Xspec

An X-Ray Spectral Fitting Package

User's Guide for version 11.2.x

Keith Arnaud and Ben Dorman
HEASARC
Laboratory for High Energy Astrophysics
NASA/GSFC
Greenbelt, MD 20771

June 2002

Contents

Chapter 1

Introduction

1.1 XSPEC

XSPEC is a command-driven, interactive, X-ray spectral-fitting program. It is designed to be completely detector-independent so that it can be used for any spectrometer. To our knowledge, XSPEC has been used to analyze data from HEAO-1 A2, *Einstein Observatory*, EXOSAT, Ginga, ROSAT, BBXRT, ASCA, CGRO, IUE, RXTE, Chandra, and XMM-Newton. It also has been used for simulations for Astro-E. This manual describes XSPEC v11.2, which we test build on Solaris 2.x, Compaq Unix (OSF), SGI/IRIX 6.5, Mac Darwin 5.4, Linux 2.2, 2.4, and 2.2ppc.

The new user is advised to read Chapter 2, which introduces spectral fitting and the XSPEC approach, Chapter 3, which gives an overview of the program commands, and Chapter 4, which contains walk-throughs of XSPEC sessions. She should then experiment with XSPEC and, if necessary, look up individual commands in Chapter 5, or descriptions of the spectral models in use, in Chapter 6. Some of the FTOOLS that can operate on XSPEC files are listed in Chapter 7. XSPEC uses the PLT plotting package, which is described briefly in Appendix A and in more detail in the “QDP/PLT User’s Guide” (Tennant, 1989). Users possessing X-ray spectra with small numbers of counts per bin are referred to Appendix B, which describes the C-statistic option. Users interested in adding their own models can read how to do so in Appendix C. The Tcl interface and XSPEC parser are described in Appendix D. Finally, Appendix E contains the recent release history.

1.2 New in Version 11.2

V11.2 is an incremental release over V11.1. Major improvements are a command to wrap-up the setting of XSPEC internal variables, methods to estimate confidence ranges on fluxes and equivalent widths, and new and improved models.

The following list summarizes the major changes to XSPEC since the previous edition of the user’s manual. Bug fixes are listed in Appendix E.

- Commands
  - → There is a new command : `xset`. This is a wrap-up for `abund`, `cosmo`, mdatadir, `method`, `statistic`, `weight`, `xsect`. It also enables the setting of strings to be passed to models.

Those currently implemented are :

APECROOT	which allows the user to change the APEC input files used in the models apec and vapec
NEIVERS	the version number for the NEI models
NEIAPECROOT	which allows the user to change the NEIAPEC input files used in the NEI models when NEIVERS=2.0.
RGS_XSOURCE_FILE	the file to read to get rgsxsrc parameters.

Note that xset does not check whether the names or strings input are valid. To use this facility for local models all you need to do is add a call to fgmstr(name) to your routine (both fgmstr and name should be declared as character*128).

→ The `eqwidth`, `flux`, and `lumin` commands now have an option to estimate a confidence range. A similar technique is used to improve the `goodness` command.

→ The `fakeit` command now automatically creates a faked background file to go with the faked source file if a background file is in use when the command is run. If the source file is called foo.bar then the background file is foo_bkg.bar.

→ `setplot rebin` has an option to change the algorithm used to calculate error bars in plots.

→ `setplot id` has an option to set a redshift.

→ `abund` (and `xset abund`) now also include the wilm relative abundances (from Wilm et al 2000 and appropriate for the ISM).

- Models

→ The NEI models now come in several versions. To switch between them use eg `xset neivers 1.0`. 1.0 gives the NEI models in XSPEC v11.1, 1.1 uses improved eigenfunction files, and 2.0 uses the improved eigenfunction files and emission spectra calculated using APEC. Emission from Ar is included in version 2.0.

→ A new model `rgsxsrc` is useful for analyzing XMM-Newton RGS spectra of extended sources.

→ The pile-up model has additional parameters for PSF fraction and number of regions. It should now be an exact analog of that available in ISIS.

- General

→ Since the Minuit source code is now freely distributable it is included in this version of XSPEC. It is no longer necessary to download an additional library in order to use the minuit options.

→ An inefficiency in reading response matrices has been removed. They now read up to five times faster than in v11.1.

→ During a fit only the variable parameters are written out instead of all the parameters.

1.3 How to contact us

XSPEC is distributed and maintained under the aegis of the GSFC High Energy Astrophysics Science Archival Research Center (HEASARC). If you have any problems with XSPEC please e-mail us at

xanprob@athena.gsfc.nasa.gov

If you want to be added to our electronic mailing list for announcements of new releases then send mail to

listserv@athena.gsfc.nasa.gov

with the main body of the message comprising the line

subscribe XANADU Your Name

The XSPEC Web page comprises links to the anonymous ftp site, a Web version of the manual, and several useful documents, including a list of known bugs. The Web address is

`http://xspec.gsfc.nasa.gov/`

1.4 Relationship to HEAsoft

HEAsoft is the amalgamated distribution of XANADU (which includes XSPEC) and FTOOLS. FTOOLS contains general purpose tools for manipulating files in FITS format. It also contains specialized tools for spectral and response files, and mission-specific tools for dealing with the data products of particular X-Ray satellites. Some of these tools are listed in Chapter 7. As well as being available as a standalone package, XSPEC is also distributed as part of the HEAsoft[1] package. If FTOOLS (V5.1) is already installed on your system, XSPEC can be built with (or linked to) the libraries provided with it. If not, the (standalone) XSPEC distribution package can be used to build those libraries. The FTOOLS package is available by anonymous ftp from

`ftp://legacy.gsfc.nasa.gov`.

Documentation for FTOOLS can be found at

http://heasarc.gsfc.nasa.gov/docs/software/ftools .

1.5 History

The first version of XSPEC was written in 1983 at the Institute of Astronomy, Cambridge, under VAX/VMS by Rick Shafer. It was written to perform spectral analysis of data from the ESA EXOSAT X-ray observatory, which was launched that year. XSPEC is a descendant of a series of spectral-fitting programs written at NASA/Goddard Space Flight Center for the OSO-8, HEAO-1 and *Einstein Observatory* missions. The initial design was the fruit of many discussions between Rick Shafer and Andy Szymkowiak.

Rick Shafer subsequently joined the EXOSAT group, where he enhanced the VAX/VMS version in collaboration with Frank Haberl. Meanwhile, Keith Arnaud ported XSPEC to a Sun/UNIX operating system. The two implementations of XSPEC diverged for several years until a determined and coordinated effort was made to recover a single version. The results of that effort was XSPECv6, described in the first edition of this manual. Subsequent editions have covered later versions of the program.

In recent years XSPEC has become the *de facto* standard for X-ray spectroscopic analysis for the ROSAT mission and the *de jure* standard for the BBXRT, ASCA, and RXTE missions. It is now in extensive use for the Chandra and XMM-Newton missions. The HEASARC, which has supported the developement of XSPEC since 1990, intends to use XSPEC for future missions as well as for those whose data are being resurrected.

The most recent published account of XSPEC can be found in Arnaud, K.A., 1996, Astronomical Data Analysis Software and Systems V, eds. Jacoby G. and Barnes J., p17, ASP Conf. Series volume 101.

[1] `http://heasarc.gsfc.nasa.gov/lheasoft`

1.6 Acknowledgements

This project would not have been possible without ignoring the advice of far more people than can be mentioned here. We would like to thank all our colleagues for their suggestions, bug reports, and (especially) source code. The GSFC X-ray astronomy group are particularly thanked for their patience exhibited while functioning as the beta test site for new releases. The initial developement of XSPEC was funded by a Royal Society grant to Prof. Andy Fabian and subsequent developement was partially funded by the European Space Agency's EXOSAT project and now by the NASA/GSFC HEASARC.

Chapter 2

Spectral Fitting and XSPEC

2.1 Introduction

This chapter is comprised of a brief description of the basics of spectral fitting, their application in XSPEC, and some helpful hints on how to approach particular problems.

2.1.1 The Basics of Spectral Fitting

Although we use a spectrometer to try to find out the spectrum of a source, what the spectrometer obtains is not the actual spectrum, but rather photon counts (C) within specific instrument channels, (I). This observed spectrum is related to the actual spectrum of the source (f(E)), such that:

$$C(I) = \int_0^\infty f(E)R(I,E)dE$$

where R(I,E) is the instrumental response and is proportional to the probability that an incoming photon of energy E will be detected in channel I. Ideally, then, we would like to determine the actual spectrum of a source, f(E), by inverting this equation, thus deriving f(E) for a given set of C(I). Regrettably, this is not possible in general, as such inversions tend to be non-unique and unstable to small changes in C(I). (For examples of attempts to circumvent these problems see Blissett & Cruise 1979; Kahn & Blissett 1980; Loredo & Epstein 1989).

The usual alternative is to try to choose a model spectrum, f(E), that can be described in terms of a few parameters (i.e., f(E,p1,p2,...)), and match, or "fit" it to the data obtained by the spectrometer. For each f(E), a predicted count spectrum ($C_p(I)$) is calculated and compared to the observed data (C(I)). Then a "fit statistic" is computed from the comparison, which enables one to judge whether the model spectrum "fits" the data obtained by the spectrometer.

The model parameters then are varied to find the parameter values that give the most desirable fit statistic. These values are referred to as the *best-fit parameters*. The model spectrum, $f_b(E)$, made up of the best-fit parameters is considered to be the *best-fit model*.

The most common fit statistic in use for determining the "best-fit" model is χ^2, defined as follows:

$$\chi^2 = \sum (C(I) - C_p(I))^2/(\sigma(I))^2$$

where $\sigma(I)$ is the error for channel I (e.g., if C(I) are counts then $\sigma(I)$ is usually estimated by $\sqrt{C(I)}$; see e.g. Wheaton et.al. 1995 for other possibilities).

Once a "best-fit" model is obtained, one must ask two questions:

1. First, one must ask how confident one can be that the observed C(I) can have been produced by the best-fit model $f_b(E)$. The answer to this question is known as the "goodness-of-fit" of the model.

   The χ^2 statistic provides a well-known goodness-of-fit criterion for a given number of degrees of freedom (ν, which is calculated as the number of channels minus the number of model parameters) and for a given confidence level. If χ^2 exceeds a critical value (tabulated in many statistics texts) one can conclude that $f_b(E)$ is *not* an adequate model for C(I). As a general rule, one wants the "reduced χ^2" (χ^2/ν) to be approximately equal to one ($\chi^2 \sim \nu$). A reduced χ^2 that is much greater than one indicates a poor fit, while a reduced χ^2 that is much less than one indicates that the errors on the data have been over-estimated.

   Even if the best-fit model ($f_b(E)$) *does* pass the "goodness-of-fit" test, one still cannot say that $f_b(E)$ is the *only* acceptable model. For example, if the data used in the fit are not particularly good, one may be able to find many different models for which adequate fits can be found. In such a case, the choice of the correct model to fit is a matter of scientific judgement.

2. Second, for a given best-fit parameter (p1), one must determine the range of values within which one can be confident the true value of the parameter lies. The answer to this question gives one the "confidence interval" for the parameter.

   The confidence interval for a given parameter is computed by varying the parameter value until the χ^2 increases by a particular amount above the minimum, or "best-fit" value.

   The amount that the χ^2 is allowed to increase (also referred to as the critical $\Delta\chi^2$) depends on the confidence level one requires, and on the number of parameters whose confidence space is being calculated. The critical $\Delta\chi^2$ for common cases are given in the following table (from Avni, 1976):

Confidence	Parameters		
	1	2	3
0.68	1.00	2.30	3.50
0.90	2.71	4.61	6.25
0.99	6.63	9.21	11.30

   There is a good discussion of confidence ranges in Press *et al.*, (1992) for readers who want more details[1]

2.2 The XSPEC implementation

To summarize the preceding section, the main components of spectral fitting are as follows:

1. The observed spectrum (C(I))
2. The instrumental response (R(I,E))
3. A model spectrum (f(E))

These components are used in the following manner:

[1] The precise reader should note that it is not correct to refer to a confidence interval as giving the probability of the true parameter value being in a given range. Such a statement is not possible in classical statistics. What is true is that if we repeated the experiment many times and calculated eg 90% confidence intervals then in 90% of the experiments the calculated confidence interval would contain the true value of the parameter.

1. A parameterized model is created that one feels represents the actual spectrum of the source.
2. One then gives values to the model parameters.
3. Based on the parameter values given, one predicts the count spectrum that would be detected by the spectrometer in a given channel for such a model.
4. Then, one compares the predicted spectrum to the spectrum actually obtained by the instrument.
5. The values of the parameters of the model are manipulated until one finds the best fit between the theoretical model and the observed data.
6. One then calculates the "goodness" of the fit to determine how well the model explains the observed data, and calculates the confidence intervals for the model's parameters.

This section describes how XSPEC performs these tasks.

2.2.1 C(I): The Observed Spectrum

To obtain the observed spectrum (C(I)), for a given observation, XSPEC uses two files: the data file, and the background file (for FITS file format see Arnaud, George & Tennant 1992[2]). The data file tells XSPEC how many total photon counts were detected by the instrument in a given channel. XSPEC then uses the background file to derive a background-subtracted C(I) in units of counts per second. The background is scaled to the data by the ratio of the BACKSCAL values in the data and background files. It also is scaled for exposure times (EXPOSURE keyword) and AREASCAL values. The background-subtracted count rate is given by :

$$C(I) = D(I)/a_D(I)/t_D - (b_D(I)/b_B(I))B(I)/a_B(I)/t_B$$

where $D(I)$ and $B(I)$ are the counts in the data and background files; t_D and t_B are the exposure times in the data and background files; $b_D(I)$ and $b_B(I)$ are the BACKSCAL values from the data and background files; and $a_D(I)$ and $a_B(I)$ are the AREASCAL values from the data and background files

When this is done XSPEC has an observed spectrum to which the model spectrum can be fit.

2.2.2 R(I,E): The Instrumental Response

Before XSPEC can take a given set of parameter values and predict the spectrum that would be detected by a given instrument, XSPEC must know the specific characteristics of the instrument. This information is known as the **detector response**. The response (R(I,E)), if you recall, is proportional to the probability that an incoming photon of energy E will be detected in channel I. As such, the response is a continuous function of E. This continuous function is converted to a discrete function by the creator of a **response matrix** who defines the energy ranges (E_J) such that:

$$R_D(I,J) = \int_{E_{J-1}}^{E_J} R(I,E)dE/(E_J - E_{J-1})$$

XSPEC reads both the energy ranges, E_J, and the response matrix ($R_D(I,J)$) from a **response file** (for FITS file format see George et al 1992[3]) in a compressed format that only stores non-zero elements.

[2] The FTOOLS distribution includes in the directory **callib/src/gen** the subroutines `rdpha2.f` and `wtpha2.f`, which can be used in programs reading and writing FITS-format spectral files

[3] The FTOOLS distribution includes in the directory **callib/src/gen** the subroutines `rdebd3.f`, `rdrmf4.f`, `wtebd3.f`, `wtrmf4.f`, which can be used in programs reading and writing FITS format response files

XSPEC also includes an option to use an **auxiliary response file** (George et al 1992[4]), which contains an array $A_D(J)$ that XSPEC multiplies into $R_D(I,J)$ as follows:

$$R_D(I,J) \rightarrow R_D(I,J) * A_D(J)$$

Conventionally, the response is in units of cm^2.

2.2.3 f(E): The Model Spectrum

The model spectrum, f(E), is calculated within XSPEC using the energy ranges defined by the response file :

$$f_D(J) = \int_{E_{J-1}}^{E_J} f(E)dE$$

and is in units of photons/cm^2/s. XSPEC allows the construction of composite models consisting of additive components (e.g., power-laws, blackbodys, and so forth), and multiplicative components, which modify additive components by an energy-dependent factor (e.g., photoelectric absorption, edges, ...). Convolution and mixing models can then perform sophisticated operations on the result. Models can be defined in algebraic notation. For example, the following expression:

$$\mathrm{phabs(power + phabs(bbody))}$$

defines an absorbed blackbody (phabs (bbody)) added to a power-law (power). The result then is modified by another absorption component (phabs).

For a more detailed explanation of models, see Chapter 6.

2.2.4 Fits and Confidence Intervals

Once data have been read in and a model defined, XSPEC uses a modified Levenberg-Marquardt algorithm (based on CURFIT from Bevington, 1969) to find the best-fit values of the model parameters. The algorithm used is a local one, so the user should be aware that it is possible for the fitting process to get stuck in a local minimum and not find the global best-fit. The process also goes much faster (and is more likely to find the true minimum) if the initial model parameters are set to sensible values.

At the end of a fit, XSPEC will write out the best-fit parameter values, along with estimated confidence intervals. These confidence intervals are one sigma and are calculated from the derivatives of the fit statistic with respect to the model parameters. However, the confidence intervals are not reliable and should be used for indicative purposes only.

XSPEC has a separate command (`error` or `uncertainty`) to derive confidence intervals for one interesting parameter, which it does by fixing the parameter of interest at a particular value and fitting for all the other parameters. New values of the parameter of interest are chosen until the appropriate delta-statistic value is obtained. XSPEC uses a bracketing algorithm followed by an iterative cubic interpolation to find the parameter value at each end of the confidence interval.

To compute confidence regions for several parameters at a time XSPEC runs a grid on these parameters. XSPEC also will display a contour plot of the confidence regions of any two parameters.

[4]The FTOOLS distribution includes in the directory **callib/src/gen** the subroutines `rdarf1.f` and `wtarf1.f`, which can be used in programs reading and writing FITS format auxiliary response files

2.3 Helpful Hints

2.3.1 Scripts and the Save command

One can write XSPEC commands into a file and have them executed by entering `@filename` at the XSPEC prompt. Also, one may enter from the shell prompt

```
% xspec - filename &
```

for batch execution (remember to end the script in file `filename` with `exit` if you want the program to terminate after completion!). Note that the default suffix for xspec scripts is `.xcm`

The `save` command writes out a file of XSPEC commands, which, when run, will restore XSPEC to the state it was in when the `save` command was issued. This command is very useful when reading a large number of data sets and/or fitting complicated models. If autosaving is operating (the default) then the equivalent of `save all xautosav.xcm` is run after each command, so if a disaster occurs it is possible to recover.

2.3.2 How to return to the XSPEC> prompt

The string /* acts as an emergency escape back to the XSPEC prompt. If one gives this string in answer to any question then one should be bounced out of whatever one is doing and returned to the XSPEC prompt. Counterexamples to this should be sent to xanprob@athena.gsfc.nasa.gov.

2.3.3 Simulations

If one has a response file for an instrument and wants to make some fake spectra, then the command `fakeit none` should be used. XSPEC will prompt the user for the response and ancillary filenames from which to build the simulated data. It is important to note that one must define a model prior to issuing the `fakeit` command.

2.3.4 Plotting Devices and Hardcopy

XSPEC uses the PGPLOT package, which comes with a standard set of device drivers. Any X-windows terminal should support /xw while xterm windows should support /xt.

The easiest way of making a hardcopy of an XSPEC plot is to use the `iplot` command and then at the PLT prompt to enter `hard /ps`. This will make a file called `pgplot.ps` which can be printed.

2.3.5 Data groups

The most common use of XSPEC is to fit one or more data sets with responses to a particular model. However, it is often useful to be able to fit simultaneously several data sets with a model whose parameters can be different for each data set. A simple example would be a number of data sets that we expect to have the same model spectrum shape but different normalizations. XSPEC caters to this need through the use of data groups. When files are read in they can be labelled as belonging to a particular data group. When a model is defined a set of model parameters is allocated for each data group. These parameters can all vary freely or they can be linked together across data groups as required.

To set up data groups, the `data` command should be given as in the following example :

```
XSPEC> data 1:1 file1 1:2 file2 2:3 file3
```

which sets up two data groups. The first data group comprises data sets from file1 and file2, and the second data group takes the data set from file3. Now when a model is defined, XSPEC will give two sets of model parameters, one for the first datagroup and one for the second.

2.3.6 What to do when you have small numbers of counts

The χ^2 statistic assumes that all the spectral channels are Gaussian distributed and that the estimate for the variance is uncorrelated with the observed counts. If there are small numbers of counts in a channel these will not be true. An alternative statistic, the C-statistic, can be used in this case. The C-statistic now in use does provide a goodness-of-fit if there are enought counts and this can be checked using the `goodness` command which uses Monte Carlo methods. This C-statistic can also provide confidence intervals in exactly the same way as χ^2. So give the command `stat cstat` and then use the `fit` and `error` commands as usual. An alternative approach is to continue using the χ^2 statistic but change the weighting to provide a better estimate of the variance in the small number limit. This can be done using the `weight gehrels` or `weight Churazov` commands. The latter is to be preferred.

2.3.7 Binning and Grouping data

Often one does not want to use the full resolution of a spectrum, either because the channels oversample the spectral resolution or because the S/N is low. XSPEC and the associated programs provide a number of ways of handling this. Firstly, the XSPEC `setplot rebin` command can be used to add channels together in the plot. It is important to realise that this effects only the plot and not the data being fitted. Two FTOOLS are available to bin and group data for fitting purposes. RBNPHA bins up the data in a non-reversible manner and should only be used to ensure that the number of bins in the spectrum is the same as that in the response. GRPPHA is the tool of choice for grouping the data to get adequate S/N or number of counts in each channel. GRPPHA does not actually add together channels, but instead sets a flag which is read by XSPEC and causes XSPEC to sum the appropriate channels. If a data file is read with some grouping then XSPEC will apply the same operation to any background or response files used.

2.3.8 How to shut XSPEC up (somewhat)

The `fit` command in XSPEC will run a certain number of iterations and then query the user whether he or she wants to continue. While useful under most circumstances, the constant questioning can be a pain if one leaves XSPEC running and expects to find it finished when one gets back, or if one habitually runs scripts. One way around this problem is to reset the number of iterations before the question is asked by giving a single argument on the `fit` command. For example, `fit 100` will run 100 iterations before asking a question. A more drastic solution is to use the `query` command. `query yes` will suppress all questions and assume that their answer is yes while `query no` will suppress all questions but assume that their answer is no.

2.3.9 References

Arnaud, K.A., George, I.M., Tennant, A.F., 1992. Legacy, 2, 65.
Avni, Y., 1976. ApJ, 210, 642.
Bevington, P.R., 1969. Data Reduction and Error Analysis for the Physical Sciences, McGraw-Hill.
Blissett, R.J., Cruise, A.M., 1979. MNRAS, 186, 45.
George, I.M., Arnaud, K.A., Pence, W., Ruamsuwan, L., 1992. Legacy, 2, 51.
Kahn, S.M., Blissett, R.J., 1980. ApJ, 238, 417.
Loredo, T.J., Epstein, R.I., 1989. ApJ, 336, 896.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 1992. Numerical Recipes (2nd edition), p687ff, CUP.
Wheaton, W.A. et.al., 1995. ApJ, 438, 322.

Chapter 3

An overview of XSPEC

3.1 Command syntax

XSPEC is a command-driven, interactive program. The user will see a prompt whenever input is required. If the user is uncertain about how to respond to a particular prompt, typing `?` usually will produce a list of valid responses.

XSPEC is designed to allow complicated, multi-instrument analysis, so most commands can take arguments specifying more than one data set. Arguments in XSPEC may be separated by either blanks or commas.

A single argument can define a range. The ranges are delimited by a dash (–). A colon (:) is used to separate ranges (e.g., the phrase 1–2:11–24 refers to channels 11–24 in files 1 and 2).

Operating-system commands can be given from within XSPEC either by giving the command name or `exec command`. A set of XSPEC commands in a file can be input using `@filename`. Note that all commands can be abbreviated to unique strings when operating in interactive mode but in scripts all commands must be specified in full. Command recall and inline editing are available.

XSPEC uses Tcl as its user interface, providing looping, conditionals, file i/o and so on. The command `tclout` can be used to pass internal XSPEC data to the Tcl level. Any Tcl scripts placed in the directory $XSPEC_HOME are auto-mounted at start-up allowing the user to create their own XSPEC commands.

3.2 Reading data

XSPEC reads in spectra from spectral files using the `data` command. Several datasets may be specified in one command. Several datasets may be stored in a single file and accessed separately. A particular data set in use may be replaced by another or dropped entirely. The input data file contains pointers to background, redistribution and auxiliary response files, but these pointers may be overridden by the `back`, `resp`, and `arf` commands. All these commands have the same syntax as `data`. PHA channels may be left out of fitting using the `ignore` command and included again using the `notice` command. These commands have a syntax allowing the same channels to be specified for more than one input file. The ignore and noticed ranges can be specified either as channels or as energies. An auxiliary background file, called the correction file, also can be included using the `corfile` command. Its use is described in the section on fitting. The current response can be replaced by a diagonal version using `diagrsp`.

3.3 Defining models

XSPEC allows users to fit data with models constructed from individual components. These components may be either additive, multiplicative, mixing, or convolution. Multiplicative components multiply the model by an energy-dependent factor. Mixing components link models for different datagroups. Convolution models modify the current model. Multiplicative, mixing, and convolution components can act on individual components, on groups of components, or on the entire model. The `model` command defines the model to be used and prompts for the starting values of its parameters. The user also can set the allowed range of the parameter. One parameter can be fixed to be a multiple of another or to be another plus a constant. Parameters can be unlinked using the `untie` command.

The value of an individual parameter can be changed with the command `newpar`. Note that `newpar 0` will print out the current parameter values. Parameters can be fixed at their current value with the `freeze` command and allowed to vary freely with the `thaw` command. Individual components can be added or subtracted from the model using `addcomp`, `delcomp`, and `editmod`. Model components can be defined interactively using `mdefine`.

The plasma emission and photoelectric absorption models require an assumption about relative elemental abundances. These relative abundances can be reset using `abund`. `xsect` can be used to switch between different compilations of photoelectric absorption cross-sections.

3.4 Fitting

The basic fit command is called `fit`. This command performs a minimization using the Levenberg-Marquardt algorithm. `fit` can take two arguments, those being 1) the number of iterations to be performed before the user is asked whether to continue; and 2) the change in fit statistic that defines convergence. The fit statistics available within XSPEC are the χ^2 and C statistics: the `statistic` command specifies which one is to be used. The `bayes` command sets up Bayesian inference. Other fit-minimization algorithms are available, and can be selected using the `method` command. A genetic algorithm is included and its operations controlled by the `genetic` command. For non-background-subtracted data the `goodness` command does a Monte Carlo calculation of the goodness-of-fit. The weighting algorithm used to calculate χ^2 can be altered by the `weight` command. A systematic model uncertainty can be included using the the `systematic` command. If the CERN library is linked in then the command `improve` can be used to try to check whether the minimum found is local or global.

The `error` or `uncertain` command calculates error bounds for one interesting parameter for the specified parameters and confidence levels. To produce multi-dimensional errors the `steppar` command is used to generate a fit-statistic grid. Two-dimensional grids may be expressed as contour plots (using `plot contour`). The model normalization can be set using the `renorm` command. The normalization of the correction file background can be set with `cornorm`, and can be set to minimize the fit statistic with `recornorm`. The gain of the response matrix can be adjusted using the `gain` command, which includes an option to fit for the gain. `ftest` provides calculation of F-test values and probabilities.

3.5 Plotting

XSPEC plotting is performed using the PLT interface. There are two basic commands: `plot` and `iplot`. The `plot` command makes a plot and returns the user to the XSPEC prompt, while the `iplot` command leaves the user in the interactive plotting interface, thus allowing the user to edit the plot. A variety of different quantities may be plotted, including the data and the current model; the integrated counts; the fit residuals; the ratio of data to model; the contributions to the fit statistic; the theoretical model; the unfolded (incident) spectrum; the detector efficiency; and the fit-statistic contours. All data plots can have an x-axis of channels, energy, or wavelength, which are specified with `setplot channel`, `setplot energy`, `setplot wavelength` respectively. The plotting device to be used is set using `setplot device` or `cpd`. Separate spectra may be added together and channels binned up (for plotting purposes only) using `setplot group` and `setplot rebin`. There is an option to plot individual additive model components on data plots, this option is enabled by `setplot add` and disabled by `setplot noadd`. Line IDs can be plotted using `setplot id` and turned off by `setplot noid`. A stack of PLT commands can be created and manipulated with `setplot command`, `setplot delete`, and `setplot list`. This command stack then is applied to every plot. A hardcopy plot can be created by issuing the command `hard /ps` at the PLT prompt.

3.6 Getting help

XSPEC has an interactive help facility that is accessed using the `help` command. Help on individual commands is available using `help commands` and on models using `help model`. If you need further help or want to send us a suggestion then send mail to

xanprob@athena.gsfc.nasa.gov

3.7 Simulated data

The `fakeit` command is used to generate simulated data. The current response matrix and model (a model must be defined prior to using the fakeit command) are used to create fake data. The user is prompted for various options. To make fake data when only a response matrix is available, give the command `fakeit none`.

3.8 Fluxes, luminosities, equivalent widths, and line IDs

The `flux` command calculates the flux from the current model in the given energy range. This energy range should be within that defined by the current response matrix. If a larger energy range is required, then the `dummyrsp` command should be given. If this command is used, the user should reset the response matrix using `resp` before returning to fitting, or incorrect results will be generated.

The `lumin` command calculates the luminosity for the source redshift given. The cosmological parameters used to calculate luminosity are changed with the `cosmo` command. The `eqwidth` command determines the equivalent width of a model component, usually a line. The command `thleqw` calculates the expected equivalent width for a fluorescence line of specified yield. The user of either of these last two commands should read the help descriptions carefully.

The command `addline` can be used to automatically add lines to a model. These can be identified using `identify` and `modid`.

3.9 Miscellaneous

XSPEC retains a memory of previous commands. These commands can be reviewed or rerun using the `history` and `!n` commands. A log of the session can be written to an ASCII file using the `log` command.

A more complete description of the session can be written to a binary file using the `xhistory` command. The `dump` command writes the current data and model into the history file. The amount of output that XSPEC writes is set by the `chatter` command, which takes two arguments applying to the terminal and to the log file.

Information on the current XSPEC status can be printed out using the `show` command. The `save` command writes the current XSPEC status to a command file, which later can be run to reset XSPEC to the same configuration. XSPEC has a mechanism to automatically save the current status and this is controlled through the `autosave` command. During the fitting procedure, XSPEC sometimes will ask the user whether he or she wishes to continue. This question can be suppressed with the `query` command. The `time` command writes out system-timing information. Finally, XSPEC can be terminated with the `exit` or `quit` commands.

Chapter 4

Walks through XSPEC

4.1 Introduction

This chapter demonstrates the use of XSPEC. The brief discussion of data and response files is followed by fully-worked examples using real data that include all the screen input and output with a variety of plots. The topics covered are as follows: defining models, fitting data, determining errors, fitting more than one set of data simultaneously, simulating data, and producing plots.

4.2 Brief Discussion of XSPEC Files

At least two files are necessary for use with XSPEC: a data file and a response file. In some cases, a file containing background may also be used, and, in rare cases, a *correction* file is needed to adjust the background during fitting. If the response is split between an rmf and an arf then an ancillary response file is also required. However, most of the time the user need only specify the data file, as the names and locations of the correct response and background files should be written in the header of the data file by whatever program created the files.

4.3 Fitting Models to Data: An Example from EXOSAT

The 6-s X-ray pulsar 1E1048.1–5937 was observed by *EXOSAT* in June 1985 for 20 ks. In this example, we'll conduct a general investigation of the spectrum from the Medium Energy (ME) instrument, i.e. follow the same sort of steps as the original investigators (Seward, Charles & Smale,1986). The ME spectrum and corresponding response matrix were obtained from the HEASARC On-line service.

Once installed, XSPEC is invoked by typing `xspec`, as in this example.

```
%xspec

               Xspec 11.1.0    21:50:14 08-Jul-2001

 For documentation, notes, and fixes see  http://xspec.gsfc.nasa.gov/

 Plot device not set, use "cpd" to set it

 Type "help" or "?" for further information
```

```
XSPEC>data s54405.pha
 Net count rate (cts/s) for file   1   3.783    +/-  0.1367
   using response (RMF) file...       s54405.rsp
   1 data set is in use
```

The `data` command tells the program to read the data as well as the response file that is named in the header of the data file. In general, XSPEC commands can be truncated, provided they remain unambiguous. Since the default extension of a data file is `.pha`, and the abbreviation of `data` to the first two letters is unambiguous, the above command can be abbreviated to `da s54405`, if desired. To obtain help on the `data` command, or on any other command, type `help command` followed by the name of the command.

One of the first things most users will want to do at this stage—even before fitting models—is to look at their data. The plotting device should be set first, with the command `cpd` (*change plotting device*). Here, we use the pgplot X-Window server, /xw.

```
XSPEC> cpd /xw
```

To see a list of alternative plot devices, type `cpd ?` There are more than 20 different things that can be plotted, all related in some way to the data, the model, the fit and the instrument. To see them, type:

```
XSPEC> plot ?
 Choose from the following 'plot' sub-commands:
 data      counts    ldata     residuals chisq     delchi    ratio
 summary   model     emodel    eemodel   contour   efficien  ufspec
 eufspec   eeufspec  dem       insensitv sensitvty genetic   foldmodel
 icounts
Insert selection: (data)
```

The most fundamental is the data plotted against instrument channel (`data`); next most fundamental, and more informative, is the data plotted against channel energy. To do this plot, use the XSPEC command `setplot energy`. Figure A shows the result of the commands:

```
XSPEC> setplot energy
XSPEC> plot data
```

People familiar with *EXOSAT* ME data will recognize the spectrum to be soft, absorbed and without an obvious bright iron emission line.

We are now ready to fit the data with a model. Models in XSPEC are specified using the `model` command, followed by an algebraic expression of a combination of model components. There are two basic kinds of model components: *additive* which represent X-Ray sources of different kinds. After being convolved with the instrument response, the components prescribe the number of counts per energy bin (e.g., a bremsstrahlung continuum); and *multiplicative* models components, which represent phenomena that modify the observed X-Radiation (e.g. reddening or an absorption edge). They apply an energy-dependent multiplicative factor to the source radiation before the result is convolved with the instrumental response.

More generally, XSPEC allows three types of modifying components: convolutions and mixing models in addition to the multiplicative type. Since there must be a source, there must be least one additive component in a model, but there is no restriction on the number of modifying components. To see what components are available, type `model ?`:

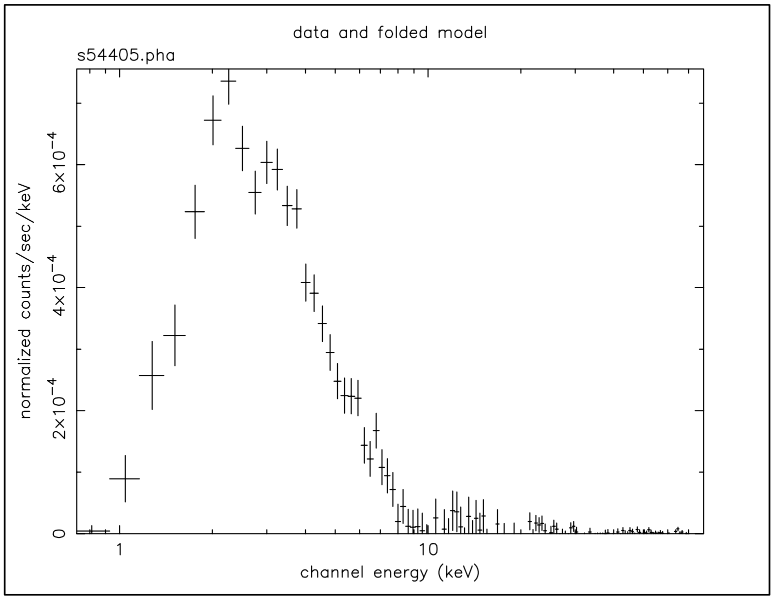

Figure A: The result of the command `plot data` when the data file in question is the *EXOSAT* ME spectrum of the 6-s X-ray pulsar 1E1048.1–5937 available from the HEASARC on-line service.

```
XSPEC>model ?

 Possible additive models are :
 apec     bbody    bbodyrad bexrav   bexriv   bknpower bmc      bremss
 c6mekl   c6pmekl  c6pvmkl  c6vmekl  cemekl   cevmkl   cflow    compbb
 compLS   compST   compTT   cutoffpl disk     diskbb   diskline diskm
 disko    diskpn   equil    gaussian gnei     grad     grbm     laor
 lorentz  meka     mekal    mkcflow  nei      npshock  nteea    pegpwrlw
 pexrav   pexriv   plcabs   powerlaw posm     pshock   raymond  redge
 refsch   sedov    srcut    sresc    step     vapec    vbremss  vequil
 vgnei    vmeka    vmekal   vmcflow  vnei     vnpshock vpshock  vraymond
 vsedov   zbbody   zbremss  zgauss   zpowerlw atable

 Possible multiplicative models are :
 absori   constant cabs     cyclabs  dust     edge     expabs   expfac
 highecut hrefl    notch    pcfabs   phabs    plabs    redden   smedge
 spline   SSS_ice  TBabs    TBgrain  TBvarabs uvred    varabs   vphabs
 wabs     wndabs   xion     zedge    zhighect zpcfabs  zphabs   zTBabs
 zvarabs  zvfeabs  zvphabs  zwabs    zwndabs  mtable   etable

 Possible mixing models are :
 ascac    projct

 Possible convolution models are :
 gsmooth  lsmooth  reflect

 Possible pile-up models are :
 pileup
```

For information about a specific component, type `help Models` followed by the name of the component):

```
XSPEC>help Models raymond

     XSPEC_11.1_commands
          Models
               raymond

An emission spectrum from hot diffuse gas based on the model calculations of
Raymond and Smith including line emission from several elements. This model
interpolates on a grid of spectra for different temperatures. The grid is
logarithmically spaced with 80 temperatures ranging from 0.008 to 80 keV.

 par1   = plasma temperature in keV
 par2   = Metal abundances (He fixed at cosmic) The elements
          included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni and
          their relative abundances are set by the abund command.
 par3   = redshift, z
 K      = 10**-14 / (4 pi (D_L/(1+z))**2) Int n_e n_H dV, where D_L is
          the luminosity distance to the source (cm), n_e is the electron
          density (cm**-3), and n_H is the hydrogen density (cm**-3)


help>
```

Given the quality of our data, as shown by the plot, we'll choose an absorbed power law, specified as follows :

```
XSPEC> model phabs(powerlaw)
```

Or, abbreviating unambiguously:

```
XSPEC> mo pha(po)
```

The user is then prompted for the initial values of the parameters. Entering <`return`> or `/` in response to a prompt uses the default values. We could also have set all parameters to their default values by entering `/*` at the first prompt. As well as the parameter values themselves, users also may specify step sizes and ranges (`value`, `delta`, `min`, `bot`, `top`, and `max values`), but here, we'll enter the defaults:

```
XSPEC>mo pha(po)
  Model:  phabs[1]( powerlaw[2] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:          1     0.001         0         0     1E+05     1E+06
phabs:nH>/*
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      1.000     +/-      0.
    2    2    2   powerlaw   PhoIndex            1.000     +/-      0.
```

```
   3    3    2   powerlaw   norm                1.000     +/-      0.
 ---------------------------------------------------------------------------
 ---------------------------------------------------------------------------
 Chi-Squared =     4.8645994E+08 using   125 PHA bins.
 Reduced chi-squared =      3987376.     for    122 degrees of freedom
 Null hypothesis probability =    0.
```

Note that most of the other numerical values in this section are have changed from those produced by earlier versions. This is because the default photoelectric absorption cross-sections have changed since XSPEC V.10. To recover identical results to earlier versions, use `xsect obcm`. `bcmc` is the default; see `help xsect` for details).

The `Current statistic` is χ^2 and is huge for the initial, default values—mostly because the power law normalization is two orders of magnitude too large. This is easily fixed using the `renorm` command :

```
XSPEC> renorm
 Chi-Squared =      852.1660     using   125 PHA bins.
 Reduced chi-squared =      6.984967     for    122 degrees of freedom
 Null hypothesis probability =    0.
```

We are not quite ready to fit the data (and obtain a better χ^2), because not all of the 125 PHA bins should be included in the fitting: some are below the lower discriminator of the instrument and therefore do not contain valid data; some have imperfect background subtraction at the margins of the pass band; and some may not contain enough counts for χ^2 to be strictly meaningful. To find out which channels to discard (*ignore* in `XSPEC` terminology), users must consult mission-specific documentation, which will inform them of discriminator settings, background subtraction problems and other issues. For the mature missions in the HEASARC archives, this information already has been encoded in the headers of the spectral files as a list of "bad" channels. Simply issue the command:

```
XSPEC> ignore bad
 Chi-Squared =      799.7109     using    85 PHA bins.
 Reduced chi-squared =      9.752572     for     82 degrees of freedom
 Null hypothesis probability =    0.
XSPEC> setplot chan
XSPEC> plot data
```

We can see that 40 channels are bad—but do we need to ignore any more? These channels are bad because of certain instrument properties: other channels still may need to be ignored because of the shape and brightness of the spectrum itself. Figure B shows the result of plotting the data in channels (using the commands `setplot channel` and `plot data`). We see that above about channel 33 the S/N becomes small. We also see, comparing Figure B with Figure A, which bad channels were ignored. Although visual inspection is not the most rigorous method for deciding which channels to ignore (more on this subject later), it's good enough for now, and will at least prevent us from getting grossly misleading results from the fitting. To ignore channels above 33:

```
XSPEC> ignore 34-**
 Chi-Squared =      677.6218     using    31 PHA bins.
 Reduced chi-squared =      24.20078     for     28 degrees of freedom
 Null hypothesis probability =    0.
```

The same result can be achieved with the command `ignore 34-125`. The inverse of `ignore` is `notice`, which has the same syntax.

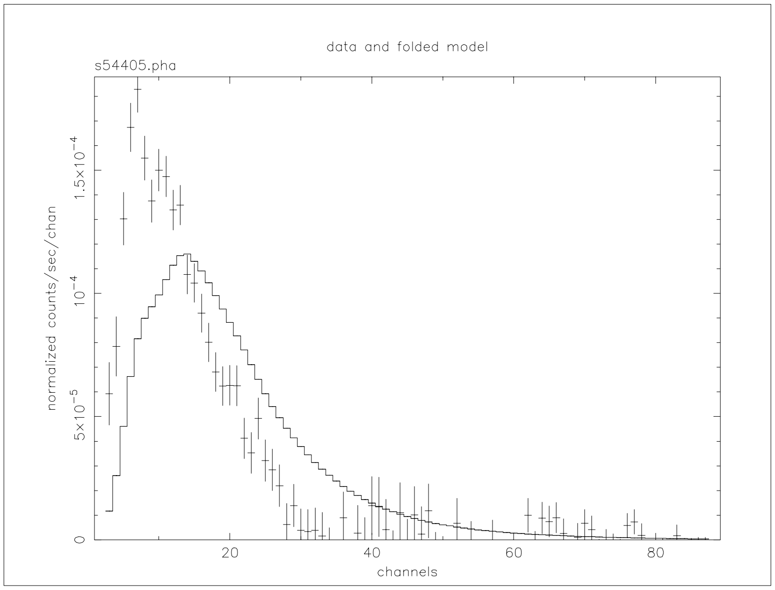

Figure B: The result of the command `plot data` after the command `ignore bad` on the *EXOSAT* ME spectrum 1E1048.1–5937.

We are now ready to fit the data. Fitting is initiated by the command `fit`. As the fit proceeds, the screen displays the status of the fit for each iteration until either the fit converges to the minimum χ^2, or the user is asked whether the fit is to go through another set of iterations to find the minimum. The default number of iterations is ten.

```
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2           3
   204.136     -3     7.9869E-02   1.564      4.4539E-03
   84.5658     -4     0.3331       2.234      1.0977E-02
   30.2511     -5     0.4422       2.174      1.1965E-02
   30.1202     -6     0.4648       2.196      1.2264E-02
   30.1189     -7     0.4624       2.195      1.2244E-02
 ---------------------------------------------------------------------------
 Variances and Principal axes :
              1     2     3
 4.14E-08 |  0.00 -0.01  1.00
 8.70E-02 | -0.91 -0.41 -0.01
 2.32E-03 | -0.41  0.91  0.01
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.4624     +/-  0.2698
    2    2    2   powerlaw   PhoIndex            2.195     +/-  0.1288
    3    3    2   powerlaw   norm               1.2244E-02 +/-  0.2415E-02
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      30.11890     using    31 PHA bins.
```

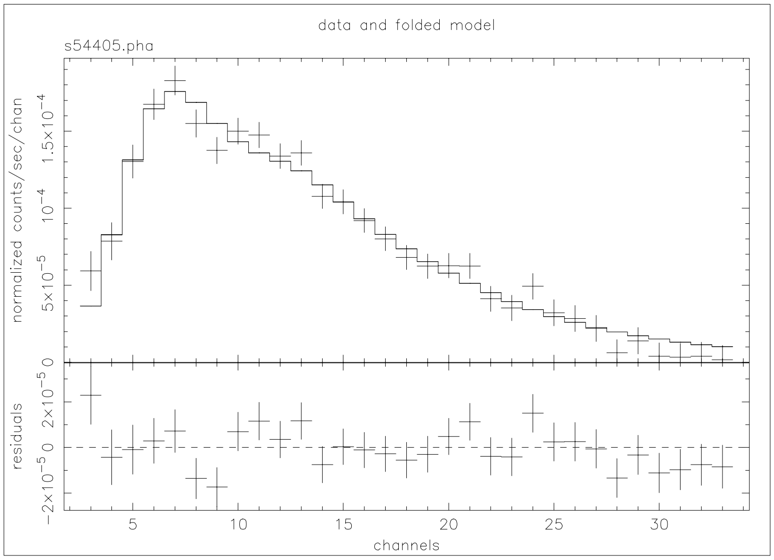

Figure C: The result of the command `plot data` with: the ME data file from 1E1048.1–5937; "bad" and negative channels ignored; the best-fitting absorbed power-law model; the residuals of the fit.

```
 Reduced chi-squared =      1.075675     for     28 degrees of freedom
 Null hypothesis probability = 0.358
```

The fit is good: reduced χ^2 is 1.075 for 31 – 3 = 28 degrees of freedom. The null hypothesis probability is the probability of getting a value of χ^2 as large or larger than observed if the model is correct. If this probability is small then the model is not a good fit. The matrix of principal axes printed out at the end of a fit provides an indication of whether parameters are correlated (at least local to the best fit). In this example the powerlaw norm is not correlated with any other parameter while the column and powerlaw index are slightly correlated.

To see the fit and the residuals, we use the command

```
XSPEC>plot data resid
```

The result is shown in Figure C.

The screen output shows the best-fitting parameter values, as well as approximations to their errors. These errors should be regarded as indications of the uncertainties in the parameters and should *not* be quoted in publications. The true errors, i.e. the confidence ranges, are obtained using the `error` command:

```
XSPEC>error 1 2 3
 Parameter   Confidence Range (     2.706)
     1    3.254765E-02    0.932778
     2     1.99397         2.40950
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
```

```
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
 *WARNING*:RENORM: No variable model to allow renormalization
     3    8.942987E-03    1.711637E-02
```

Here, the numbers 1, 2, 3 refer to the parameter numbers in the `Model par` column of the screen output. For the first parameter, the column of absorbing hydrogen atoms, the 90% confidence range is $3.0 \times 10^{20} < N_H < 8.6 \times 10^{21}$ cm^{-2}. This corresponds to an excursion in χ^2 of 2.706. The reason these “better” errors are not given automatically as part of the `fit` output is that they entail further fitting. When the model is simple, this does not require much CPU, but for complicated models the extra time can be considerable. The warning message is generated because there are no free normalizations in the model while the error is being calculated on the normalization itself. In this case, the warning may safely be ignored.

What else can we do with the fit? One thing is to derive the flux of the model. The data by themselves only give the instrument-dependent count rate. The model, on the other hand, is an estimate of the true spectrum emitted. In XSPEC, the model is defined in physical units independent of the instrument. The command `flux` integrates the current model over the range specified by the user:

```
XSPEC> flux 2 10
 Model flux  3.5496E-03 photons ( 2.2492E-11 ergs)cm**-2 s**-1 (  2.000- 10.000)
```

Here, we have chosen the standard X-ray range of 2–10 keV and find that the energy flux is 2.2×10^{-11} erg cm^{-2} s^{-1}. Note that `flux` will integrate only within the energy range of the current response matrix. If the model flux outside this range is desired—in effect, an extrapolation beyond the data—then the command `dummyrsp` should be used. This command sets up a dummy response that covers the range required. For example, if we want to know the flux of our model in the *ROSAT* PSPC band of 0.2–2 keV, we enter:

```
XSPEC>dummy 0.2 2.
 Chi-Squared =      3583.779     using    31 PHA bins.
 Reduced chi-squared =      127.9921     for     28 degrees of freedom
 Null hypothesis probability =    0.
XSPEC>flux 0.2 2.
 Model flux  4.5306E-03 photons ( 9.1030E-12 ergs)cm**-2 s**-1 (  0.200-  2.000)
```

The energy flux, at 9.1×10^{-12} erg cm^{-2} s^{-1}, is lower in this band but the photon flux is higher. To get our original response matrix back we enter :

```
XSPEC> response
 Chi-Squared =      30.11890     using    31 PHA bins.
 Reduced chi-squared =      1.075675     for     28 degrees of freedom
 Null hypothesis probability = 0.358
```

The fit, as we’ve remarked, is good, and the parameters are constrained. But unless the purpose of our investigation is merely to measure a photon index, it’s a good idea to check whether alternative models can fit the data just as well. We also should derive upper limits to components such as iron emission lines and additional continua, which, although not evident in the data nor required for a good fit, are nevertheless important to constrain. First, let’s try an absorbed black body:

```
XSPEC>mo pha(bb)
  Model:  phabs[1]( bbody[2] )
Input parameter value, delta, min, bot, top, and max values for ...
```

```
Current:          1     0.001        0         0     1E+05     1E+06
phabs:nH>/*
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( bbody[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      1.000     +/-      0.
    2    2    2   bbody      kT       keV        3.000     +/-      0.
    3    3    2   bbody      norm                1.000     +/-      0.
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =     3.3142067E+09 using    31 PHA bins.
 Reduced chi-squared =     1.1836453E+08 for     28 degrees of freedom
 Null hypothesis probability =    0.
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2           3
   1420.96      0     0.2116       2.987      7.7666E-04
   1387.72      0     4.4419E-02   2.975      7.7003E-04
   1376.39      0     4.3009E-03   2.963      7.6354E-04
   1371.67      0     1.8192E-03   2.951      7.5730E-04
   1367.04      0     5.8100E-04   2.939      7.5130E-04
   1362.47      0     1.9059E-04   2.926      7.4548E-04
   1357.84      0     6.1455E-05   2.913      7.3982E-04
   1353.14      0     2.5356E-05   2.900      7.3429E-04
   1348.36      0     6.7582E-06   2.887      7.2885E-04
   1343.50      0     2.0479E-06   2.874      7.2350E-04
 Number of trials exceeded - last iteration delta =    4.861
 Continue fitting? (Y)y
 ...

   113.954      0         0.      0.8907      2.7865E-04
   113.954     -1         0.      0.8905      2.7859E-04
   113.954      4         0.      0.8905      2.7859E-04
 ---------------------------------------------------------------------------
 Variances and Principal axes :
              2     3
 2.88E-04 | -1.00  0.00
 8.45E-11 |  0.00 -1.00
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( bbody[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22         0.     +/-  -1.000
    2    2    2   bbody      kT       keV       0.8905     +/-  0.1696E-01
    3    3    2   bbody      norm               2.7859E-04 +/-  0.9268E-05
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      113.9542     using    31 PHA bins.
 Reduced chi-squared =      4.069792     for     28 degrees of freedom
 Null hypothesis probability = 2.481E-12
```

Note that when more than 10 iterations are required for convergence the user is asked whether or not to

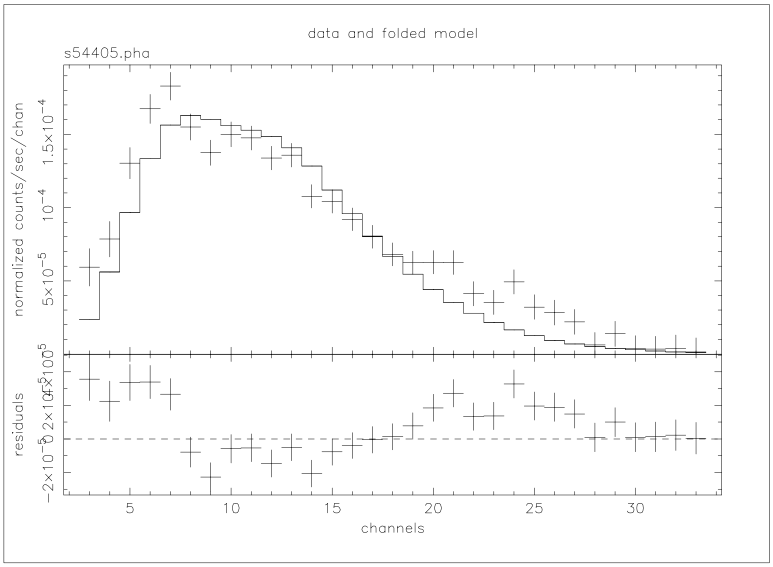

Figure D: As for Figure C, but the model is the best-fitting absorbed black body. Note the wave-like shape of the residuals which indicates how poor the fit is, i.e. that the continuum is obviously *not* a black body.

continue at the end of each set of 10. Saying *no* at these prompts is a good idea if the fit is not converging quickly. Conversely, to avoid having to keep answering the question, i.e., to increase the number of iterations before the prompting question appears, begin the fit with, say `fit 100`. This command will put the fit through 100 iterations before pausing.

Plotting the data and residuals again with

```
XSPEC> plot data resid
```

we obtain Figure D.

The black body fit is obviously not a good one. Not only is χ^2 large, but the best-fitting N_H is rather low. Inspection of the residuals confirms this: the pronounced wave-like shape is indicative of a bad choice of overall continuum (see Figure D). Let's try thermal bremsstrahlung next:

```
XSPEC>mo pha(br)
  Model:  phabs[1]( bremss[2] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:          1     0.001         0         0     1E+05     1E+06
phabs:nH>/*
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( bremss[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      1.000     +/-      0.
    2    2    2   bremss     kT       keV        7.000     +/-      0.
    3    3    2   bremss     norm                1.000     +/-      0.
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
```

```
 Chi-Squared =     4.5311800E+07 using    31 PHA bins.
 Reduced chi-squared =      1618279.     for     28 degrees of freedom
 Null hypothesis probability =    0.
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2           3
   113.305     -3     0.2441       6.557      6.8962E-03
   40.4519     -4     0.1173       5.816      7.7944E-03
   36.0549     -5     4.4750E-02   5.880      7.7201E-03
   33.4168     -6     1.8882E-02   5.868      7.7476E-03
   32.6766     -7     7.8376E-03   5.864      7.7495E-03
   32.3192     -8     2.7059E-03   5.862      7.7515E-03
   32.1512     -9     2.3222E-04   5.861      7.7525E-03
   32.1471    -10     1.0881E-04   5.861      7.7523E-03
 ---------------------------------------------------------------------------
 Variances and Principal axes :
              1     2     3
 2.29E-08 |  0.00  0.00  1.00
 3.18E-02 |  0.95  0.31  0.00
 8.25E-01 |  0.31 -0.95  0.00
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( bremss[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     1.0881E-04 +/-  0.3290
    2    2    2   bremss     kT       keV        5.861     +/-  0.8651
    3    3    2   bremss     norm               7.7523E-03 +/-  0.8122E-03
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      32.14705     using    31 PHA bins.
 Reduced chi-squared =      1.148109     for     28 degrees of freedom
 Null hypothesis probability = 0.269
```

Bremsstrahlung is a better fit than the black body—and is as good as the power law—although it shares the low N_H. With two good fits, the power law and the bremsstrahlung, it's time to scrutinize their parameters in more detail.

First, we reset our fit to the powerlaw (output omitted):

```
XSPEC>mo pha(po)
```

From the *EXOSAT* database on HEASARC, we know that the target in question, 1E1048.1–5937, has a Galactic latitude of 24 arcmin, i.e., almost on the plane of the Galaxy. In fact, the data base also provides the value of the Galactic N_H based on 21-cm radio observations. At 4×10^{22} cm^{-2}, it is higher than the 90 percent-confidence upper limit from the power-law fit. Perhaps, then, the power-law fit is not so good after all. What we can do is fix (*freeze* in XSPEC terminology) the value of N_H at the Galactic value and refit the power law. Although we won't get a good fit, the shape of the residuals might give us a clue to what is missing. To freeze a parameter in XSPEC, use the command `freeze` followed by the parameter number, like this:

```
XSPEC> freeze 1
 Number of variable fit parameters =    2
```

The inverse of `freeze` is `thaw`:

```
XSPEC> thaw 1
 Number of variable fit parameters =    3
```

Alternatively, parameters can be frozen using the `newpar` command, which allows all the quantities associated with a parameter to be changed. The second quantity, `delta`, is the step size used to calculate the derivative in the fitting, and, if set to a negative number, will cause the parameter to be frozen. In our case, we want N_H frozen at 4×10^{22} cm^{-2}, so we go back to the power law best fit and do the following :

```
XSPEC>newpar 1
Current:      0.463     0.001         0         0     1E+05     1E+06
phabs:nH>4,0
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      4.000     frozen
    2    2    2   powerlaw   PhoIndex            2.195     +/-  0.1287
    3    3    2   powerlaw   norm               1.2247E-02 +/-  0.2412E-02
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
    2 variable fit parameters
 Chi-Squared =      829.3545     using    31 PHA bins.
 Reduced chi-squared =      28.59843     for     29 degrees of freedom
 Null hypothesis probability =    0.
```

Note the useful trick of giving a value of zero for `delta` in the `newpar` command. This has the effect of changing `delta` to the negative of its current value. If the parameter is free, it will be frozen, and if frozen, thawed. The same result can be obtained by putting everything onto the command line, i.e., `newpar 1 4, 0`, or by issuing the two commands, `newpar 1 4` followed by `freeze 1`. Now, if we fit and plot again, we get the following model (Fig. E).

```
XSPEC>fit
...
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      4.000     frozen
    2    2    2   powerlaw   PhoIndex            3.594     +/-  0.6867E-01
    3    3    2   powerlaw   norm               0.1161     +/-  0.9412E-02
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      125.5134     using    31 PHA bins.
 Reduced chi-squared =      4.328048     for     29 degrees of freedom
 Null hypothesis probability = 5.662E-14
 XSPEC>plot data resid
```

The fit is not good. In Figure E we can see why: there appears to be a surplus of softer photons, perhaps indicating a second continuum component. To investigate this possibility we can add a component to our model. The `editmod` command lets us do this without having to respecify the model from scratch. Here, we'll add a black body component.

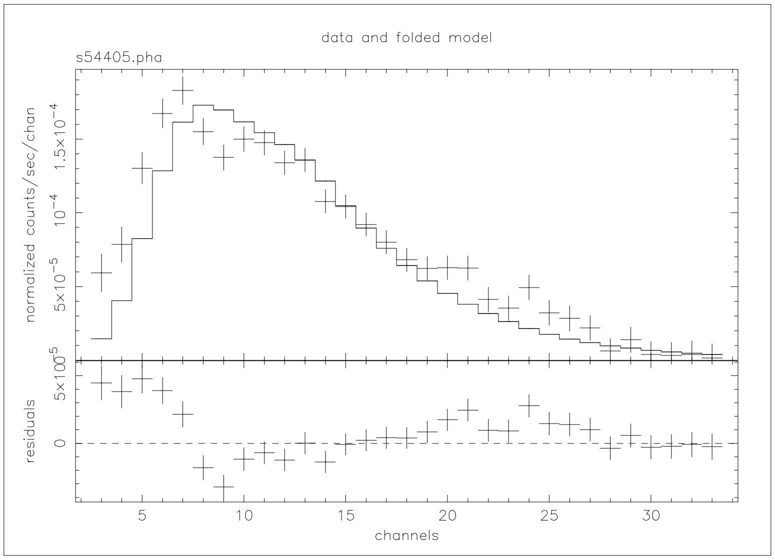

Figure E: As for Figure C & D, but the model is the best-fitting power law with the absorption fixed at the Galactic value. Under the assumptions that the absorption really is the same as the 21-cm value and that the continuum really is a power law, this plot provides some indication of what other components might be added to the model to improve the fit.

```
XSPEC>editmod pha(po+bb)
  Model:  phabs[1]( powerlaw[2] + bbody[3] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:         3      0.01    0.0001      0.01       100       200
bbody:kT>2,0
Current:         1      0.01         0         0     1E+24     1E+24
bbody:norm>1.e-5
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] + bbody[3] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      4.000     frozen
    2    2    2   powerlaw   PhoIndex            3.594     +/-  0.6867E-01
    3    3    2   powerlaw   norm               0.1161     +/-  0.9412E-02
    4    4    3   bbody      kT       keV        2.000     frozen
    5    5    3   bbody      norm               1.0000E-05 +/-      0.
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      122.1538     using    31 PHA bins.
 Reduced chi-squared =      4.362635     for     28 degrees of freedom
 Null hypothesis probability = 9.963E-14
```

Notice that in specifying the initial values of the black body, we have frozen kT at 2 keV (the canonical temperature for nuclear burning on the surface of a neutron star in a low-mass X-ray binary) and started the normalization at zero. Without these measures, the fit might “lose its way”. Now, if we fit, we get (not showing all the iterations this time):

```
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] + bbody[3] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      4.000     frozen
    2    2    2   powerlaw   PhoIndex            4.932     +/-  0.1618
    3    3    2   powerlaw   norm               0.3761     +/-  0.5449E-01
    4    4    3   bbody      kT       keV        2.000     frozen
    5    5    3   bbody      norm               2.3212E-04 +/-  0.3966E-04
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      55.63374     using    31 PHA bins.
 Reduced chi-squared =      1.986919     for     28 degrees of freedom
 Null hypothesis probability = 1.425E-03
```

The fit is better than the one with just a power law *and* the fixed Galactic column, but it is still not good. Thawing the black body temperature and fitting gives us:

```
XSPEC>thaw 4
 Number of variable fit parameters =    4
XSPEC>fit
...
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] + bbody[3] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22      4.000     frozen
    2    2    2   powerlaw   PhoIndex            6.401     +/-  0.3873
    3    3    2   powerlaw   norm                1.086     +/-  0.3032
    4    4    3   bbody      kT       keV        1.199     +/-  0.8082E-01
    5    5    3   bbody      norm               2.6530E-04 +/-  0.3371E-04
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      37.21207     using    31 PHA bins.
 Reduced chi-squared =      1.378225     for     27 degrees of freedom
 Null hypothesis probability = 9.118E-02
```

This, of course, is a better fit, but the photon index of the power law has ended up extremely and implausibly steep. Looking at this odd model with the command

```
XSPEC> plot model
```

we see, in Figure F, that the black body and the power law have changed places, in that the power law provides the soft photons required by the high absorption, while the black body provides the harder photons.

We could continue to search for a plausible, well-fitting model, but the data, with their limited signal-to-noise and energy resolution, probably don't warrant it (the original investigators published only the power law fit). There is, however, one final, useful thing to do with the data: derive an upper limit to the presence of a fluorescent iron emission line. First we delete the black body component using `delcomp`:

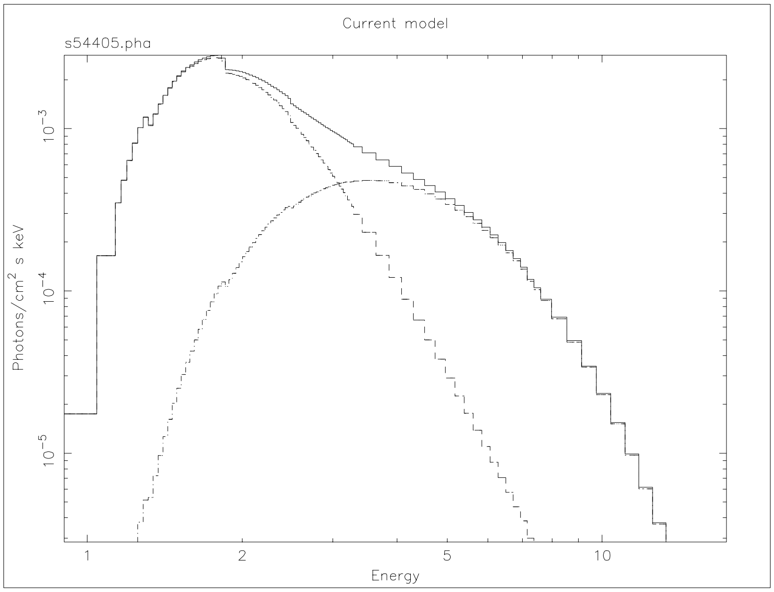

Figure F: The result of the command `plot model` in the case of the ME data file from 1E1048.1–5937. Here, the model is the best-fitting combination of power law, black body and fixed Galactic absorption. The three lines show the two continuum components (absorbed to the same degree) and their sum.

```
XSPEC>delcomp 3
  Model:  phabs[1]( powerlaw[2] )
 Chi-Squared =      1285.487     using    31 PHA bins.
 Reduced chi-squared =      44.32712     for     29 degrees of freedom
 Null hypothesis probability =    0.
```

Then we thaw N_H and refit to recover our original, best fit:

```
XSPEC>thaw 1
 Number of variable fit parameters =    3
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2           3
   924.178     -2      5.087       5.076      0.4056
   305.507     -2      4.525       3.791      0.1249
   140.460     -2      2.930       3.367      6.5553E-02
   64.4275     -3     0.6068       2.244      1.4635E-02
   30.3738     -4     0.4837       2.201      1.2279E-02
   30.1189     -5     0.4641       2.195      1.2258E-02
   30.1189     -6     0.4637       2.195      1.2255E-02
 ---------------------------------------------------------------------------
 Variances and Principal axes :
              1     2     3
 4.13E-08 |  0.00 -0.01  1.00
 8.69E-02 | -0.91 -0.41 -0.01
 2.31E-03 | -0.41  0.91  0.01
  ---------------------------------------------------------------------------
```

```
  -----------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.4637     +/-  0.2696
    2    2    2   powerlaw   PhoIndex            2.195     +/-  0.1287
    3    3    2   powerlaw   norm               1.2255E-02 +/-  0.2412E-02
  -----------------------------------------------------------------------
  -----------------------------------------------------------------------
 Chi-Squared =      30.11890     using    31 PHA bins.
 Reduced chi-squared =      1.075675     for     28 degrees of freedom
 Null hypothesis probability = 0.358
```

Now, we add a gaussian emission line of fixed energy and width:

```
XSPEC>editmod pha(po+ga)
  Model:  phabs[1]( powerlaw[2] + gaussian[3] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:        6.5      0.05         0         0     1E+06     1E+06
gaussian:LineE>6.4 0
Current:        0.1      0.05         0         0        10        20
gaussian:Sigma>0.1 0
Current:          1      0.01         0         0     1E+24     1E+24
gaussian:norm>1.e-4
  -----------------------------------------------------------------------
  -----------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] + gaussian[3] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.4637     +/-  0.2696
    2    2    2   powerlaw   PhoIndex            2.195     +/-  0.1287
    3    3    2   powerlaw   norm               1.2255E-02 +/-  0.2412E-02
    4    4    3   gaussian   LineE    keV        6.400     frozen
    5    5    3   gaussian   Sigma    keV       0.1000     frozen
    6    6    3   gaussian   norm               1.0000E-04 +/-      0.
  -----------------------------------------------------------------------
  -----------------------------------------------------------------------
 Chi-Squared =      32.75002     using    31 PHA bins.
 Reduced chi-squared =      1.212964     for     27 degrees of freedom
 Null hypothesis probability = 0.205
XSPEC>fit
...
  -----------------------------------------------------------------------
  -----------------------------------------------------------------------
  Model:  phabs[1]( powerlaw[2] + gaussian[3] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.6562     +/-  0.3193
    2    2    2   powerlaw   PhoIndex            2.324     +/-  0.1700
    3    3    2   powerlaw   norm               1.4636E-02 +/-  0.3642E-02
    4    4    3   gaussian   LineE    keV        6.400     frozen
    5    5    3   gaussian   Sigma    keV       0.1000     frozen
    6    6    3   gaussian   norm               9.6462E-05 +/-  0.9542E-04
  -----------------------------------------------------------------------
```

```
 ---------------------------------------------------------------------------
 Chi-Squared =      29.18509     using    31 PHA bins.
 Reduced chi-squared =      1.080929     for     27 degrees of freedom
 Null hypothesis probability = 0.352
```

The energy and width have to be frozen because,in the absence of an obvious line in the data, the fit would be completely unable to converge on meaningful values. Besides, our aim is to see how bright a line at 6.4 keV can be and still not ruin the fit. To do this, we fit first and then use the `error` command to derive the maximum allowable iron line normalization. We then set the normalization at this maximum value with `newpar` and, finally, derive the equivalent width using the `eqwidth` command. That is:

```
XSPEC>err 6
 Parameter   Confidence Range (     2.706)
 Parameter pegged at hard limit         0.
 with delta ftstat=    0.9338
     6          0.        1.530722E-04
XSPEC>new 6  1.530722E-04
    4 variable fit parameters
 Chi-Squared =      34.91923     using    31 PHA bins.
 Reduced chi-squared =      1.293305     for     27 degrees of freedom
 Null hypothesis probability = 0.141
XSPEC>eqwidth 3
 Additive group equiv width for model 3 (gaussian): 839. eV
```

Things to note:

- The true minimum value of the gaussian normalization is less than zero, but the `error` command stopped searching for a $\Delta\chi^2$ of 2.706 when the minimum value hit zero, the "hard" lower limit of the parameter. Hard limits can be adjusted with the `newpar` command, and they correspond to the quantities `min` and `max` associated with the parameter values. In fact, according to the screen output, the value of $\Delta\chi^2$ corresponding to zero normalization is 0.934.

- The command `eqwidth` takes the component number as its argument.

- The upper limit on the equivalent width of a 6.4 keV emission line is high (839 eV)!

4.4 Simultaneous Fitting: Examples from Einstein and Ginga

XSPEC has the very useful facility of allowing models to be fitted simultaneously to more than one data file. It is even possible to group files together and to fit different models simultaneously. Reasons for fitting in this manner include:

- *The same target is observed at several epochs but, although the source stays constant, the response matrix has changed.* When this happens, the data files cannot be added together; they have to be fitted separately. Fitting the data files simultaneously yields tighter constraints.

- *The same target is observed with different instruments.* The GIS and SIS on *ASCA*, for example, observe in the same direction simultaneously. As far as XSPEC is concerned, this is just like the previous case: two data files with two responses fitted simultaneously with the same model.

- *Different targets are observed, but the user wants to fit the same model to each data file with some parameters shared and some allowed to vary separately.* For example, if you have a series of spectra from a variable AGN, you might want to fit them simultaneously with a model that has the same, common photon index but separately vary the normalization and absorption.

Other scenarios are possible—the important thing is to recognize the flexibility of XSPEC in this regard.

As an example of the first case, we'll fit two spectra derived from two separate *Einstein* Solid State Spectrometer (SSS) observations of the cooling-flow cluster Abell 496. Although the two observations were carried out on consecutive days (in August 1979), the response is different, due to the variable build-up of ice on the detector. This problem bedeviled analysis during the mission; however, it has now been calibrated successfully and is incorporated into the response matrices associated with the spectral files in the HEASARC archive. The SSS also provides an example of how background correction files are used in XSPEC.

To fit the same model with the same set of parameters to more than one data file, simply enter the names of the data files after the `data` command:

```
XSPEC> data sa496b.pha sa496c.pha
 Net count rate (cts/s) for file   1  0.7806    +/-  9.3808E+05( 86.9% total)
   using response (RMF) file...       sa496b.rsp
   using background file...           sa496b.bck
   using correction file...           sa496b.cor
 Net count rate (cts/s) for file   2  0.8002    +/-  9.3808E+05( 86.7% total)
   using response (RMF) file...       sa496c.rsp
   using background file...           sa496c.bck
   using correction file...           sa496c.cor
 Net correction flux for file  1=    8.4469E-04
 Net correction flux for file  2=    8.7577E-04
   2 data sets are in use
```

As the messages indicate, XSPEC also has read in the associated:

- response files (`sa496b.rsp` & `sa496c.rsp`),
- background files (`sa496b.bck` & `sa496c.bck`) and
- correction files (`sa496b.cor` & `sa496c.cor`).

These files are all listed in the headers of the data files (`sa496b.pha` & `sa496c.pha`).

To ignore channels, the file number (1 & 2 in this example) precedes the range of channels to be ignored. Here, we wish to ignore, for both files, channels 1–15 and channels 100–128. This can be done by specifying the files one after the other with the range of ignored channels:

```
XSPEC> ignore 1:1-15 1:100-128 2:1-15 2:100-128
 Chi-Squared =      1933.559     using   168 PHA bins.
 Reduced chi-squared =      11.79000     for    164 degrees of freedom
 Null hypothesis probability =    0.
```

or by specifying the range of file number with the channel range:

```
XSPEC> ignore 1-2:1-15 100-128
```

In this example, we'll fit a cooling-flow model under the reasonable assumption that the small SSS field of view sees mostly just the cool gas in the middle of the cluster. We'll freeze the values of the maximum temperature (the temperature from which the gas cools) and of the abundance to the values found by instruments such as the *Ginga* LAC and the *EXOSAT* ME, which observed the entire cluster. The minimum gas temperature is frozen at 0.1 keV; the "slope" is frozen at zero (isobaric cooling) and the normalization is given an initial value of 100 solar masses per year:

```
XSPEC>mo pha(cflow)
  Model:  phabs[1]( cflow[2] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:         1     0.001        0         0     1E+05     1E+06
phabs:nH>0.045
Current:         0      0.01       -5        -5         5         5
cflow:slope>0,0
Current:       0.1     0.001   0.0808    0.0808      79.9      79.9
cflow:lowT>0.1,0
Current:         4     0.001   0.0808    0.0808      79.9      79.9
cflow:highT>4,0
Current:         1      0.01        0         0         5         5
cflow:Abundanc>0.5,0
Current:         0      -0.1        0         0       100       100
cflow:Redshift>0.032
Current:         1      0.01        0         0     1E+24     1E+24
cflow:norm>100
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( cflow[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     4.5000E-02 +/-      0.
    2    2    2   cflow      slope                 0.     frozen
    3    3    2   cflow      lowT     keV       0.1000    frozen
    4    4    2   cflow      highT    keV        4.000    frozen
    5    5    2   cflow      Abundanc           0.5000    frozen
    6    6    2   cflow      Redshift           3.2000E-02 frozen
    7    7    2   cflow      norm                100.0    +/-      0.
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      2740.606     using   168 PHA bins.
 Reduced chi-squared =      16.50968     for    166 degrees log none
m
 Null hypothesis probability =    0.
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2          3           4
                 5           6           7
   414.248     -3     0.2050          0.      0.1000       4.000
              0.5000      3.2000E-02   288.5
   373.205     -4     0.2508          0.      0.1000       4.000
              0.5000      3.2000E-02   321.9
   372.649     -5     0.2566          0.      0.1000       4.000
              0.5000      3.2000E-02   325.9
   372.640     -6     0.2574          0.      0.1000       4.000
              0.5000      3.2000E-02   326.3
 ---------------------------------------------------------------------------
```

```
 Variances and Principal axes :
               1     7
 3.55E-05 | -1.00  0.00
 3.52E+01 |  0.00 -1.00
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( cflow[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.2574     +/-  0.9219E-02
    2    2    2   cflow      slope                  0.     frozen
    3    3    2   cflow      lowT     keV       0.1000     frozen
    4    4    2   cflow      highT    keV        4.000     frozen
    5    5    2   cflow      Abundanc           0.5000     frozen
    6    6    2   cflow      Redshift           3.2000E-02 frozen
    7    7    2   cflow      norm                326.3     +/-   5.929
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      372.6400     using   168 PHA bins.
 Reduced chi-squared =      2.244819     for    166 degrees of freedom
 Null hypothesis probability = 6.535E-18
```

As we can see, χ^2 is not good, but the high statistic could be because we have yet to adjust the correction file. Correction files in XSPEC take into account detector features that cannot be completely prescribed *ab initio* and which must be fitted at the same time as the model. *Einstein* SSS spectra, for example, have a background feature the level of which varies unpredictably. Its spectral form is contained in the correction file, but its normalization is determined by fitting. This fitting is set in motion using the command `recornrm` (*reset the correction-file normalization*):

```
XSPEC>reco 1
  File #     Correction
      1      0.4118 +/-  0.0673
 After correction norm adjustment  0.412 +/- 0.067
 Chi-Squared =      335.1577     using   168 PHA bins.
 Reduced chi-squared =      2.019022     for    166 degrees of freedom
 Null hypothesis probability = 1.650E-13
XSPEC>reco 2
  File #     Correction
      2      0.4864 +/-  0.0597
 After correction norm adjustment  0.486 +/- 0.060
 Chi-Squared =      268.8205     using   168 PHA bins.
 Reduced chi-squared =      1.619400     for    166 degrees of freedom
 Null hypothesis probability = 7.552E-07
```

This process is iterative, and, in order to work, must be used in tandem with fitting the model. Successive fits and recorrections are applied until the fit is stable, i.e., until further improvement in χ^2 no longer results. Of course, this procedure is only worthwhile when the model gives a reasonably good account of the data. Eventually, we end up at:

```
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2           3           4
               5           6           7
   224.887     -3     0.2804          0.      0.1000       4.000
```

```
              0.5000      3.2000E-02   313.0
   224.792     -4    0.2835          0.      0.1000       4.000
              0.5000      3.2000E-02   314.5
   224.791     -5    0.2837          0.      0.1000       4.000
              0.5000      3.2000E-02   314.6
 ---------------------------------------------------------------------------
 Variances and Principal axes :
               1     7
 4.64E-05 | -1.00  0.00
 3.78E+01 |  0.00 -1.00
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( cflow[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.2837     +/-  0.1051E-01
    2    2    2   cflow      slope                 0.      frozen
    3    3    2   cflow      lowT     keV       0.1000     frozen
    4    4    2   cflow      highT    keV        4.000     frozen
    5    5    2   cflow      Abundanc           0.5000     frozen
    6    6    2   cflow      Redshift           3.2000E-02 frozen
    7    7    2   cflow      norm                314.6     +/-   6.147
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      224.7912     using   168 PHA bins.
 Reduced chi-squared =      1.354164     for    166 degrees of freedom
 Null hypothesis probability = 1.616E-03
```

The final value of χ^2 is much better than the original, but is not quite acceptable. However, the current model has only two free parameters: further explorations of parameter space would undoubtedly improve the fit.

We'll leave this example and move on to look at another kind of simultaneous fitting: one where the same model is fitted to two different data files. This time, not all the parameters will be identical. The massive X-ray binary Centaurus X-3 was observed with the LAC on *Ginga* in 1989. Its flux level before eclipse was much lower than the level after eclipse. Here, we'll use XSPEC to see whether spectra from these two phases can be fitted with the same model, which differs only in the amount of absorption. This kind of fitting relies on introducing an extra dimension, the *group*, to the indexing of the data files. The files in each group share the same model but not necessarily the same parameter values, which may be shared as common to all the groups or varied separately from group to group. Although each group may contain more than one file, there is only one file in each of the two groups in this example. Groups are specified with the `data` command, with the group number preceding the file number, like this:

```
XSPEC> da 1:1 losum 2:2 hisum
 Net count rate (cts/s) for file   1   140.1    +/-  0.3549
   using response (RMF) file...       ginga_lac.rsp
 Net count rate (cts/s) for file   2   1371.    +/-   3.123
   using response (RMF) file...       ginga_lac.rsp
   2 data sets are in use
```

Here, the first group makes up the file `losum.pha`, which contains the spectrum of all the low, pre-eclipse emission. The second group makes up the second file, `hisum.pha`, which contains all the high, post-eclipse emission. Note that file number is "absolute" in the sense that it is independent of group number.

Thus, if there were three files in each of the two groups (lo1.pha, lo2.pha, lo3.pha, hi1.pha, hi2.pha & hi3.pha, say), rather than one, the six files would be specified as

```
da 1:1 lo1 1:2 lo2 1:3 lo3 2:4 hi1 2:5 hi2 2:6 hi3
```

The ignore command works, as usual, on file number, and does not take group number into account. So, to ignore channels 1–3 and 37–48 of both files:

```
XSPEC> ignore 1-2:1-3 37-48
```

The model we'll use at first to fit the two files is an absorbed power law with a high-energy cut-off:

```
XSPEC> mo phabs * highecut (po)
```

After defining the model, the user is prompted for two sets of parameter values, one for the first group of data files (losum.pha), the other for the second group (hisum.pha). Here, we'll enter the absorption column of the first group as 10^{24} cm^{-2} and enter the default values for all the other parameters in the first group. Now, when it comes to the second group of parameters, we enter a column of 10^{22} cm^{-2} and then enter defaults for the other parameters. The rule being applied here is as follows: to tie parameters in the second group to their equivalents in the first group, take the default when entering the second-group parameters; to allow parameters in the second group to vary independently of their equivalents in the first group, enter different values explicitly:

```
XSPEC>mo phabs*highecut(po)
  Model:  phabs[1]*highecut[2]( powerlaw[3] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:          1     0.001         0         0     1E+05     1E+06
DataGroup 1:phabs:nH>100
Current:         10      0.01    0.0001      0.01     1E+06     1E+06
DataGroup 1:highecut:cutoffE>/
Current:         15      0.01    0.0001      0.01     1E+06     1E+06
DataGroup 1:highecut:foldE>/
Current:          1      0.01        -3        -2         9        10
DataGroup 1:powerlaw:PhoIndex>/
Current:          1      0.01         0         0     1E+24     1E+24
DataGroup 1:powerlaw:norm>/
Current:        100     0.001         0         0     1E+05     1E+06
DataGroup 2:phabs:nH>1
Current:         10      0.01    0.0001      0.01     1E+06     1E+06
DataGroup 2:highecut:cutoffE>/*
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]*highecut[2]( powerlaw[3] )
  Model Fit Model Component  Parameter  Unit     Value                   Data
  par   par comp                                                         group
    1    1    1   phabs      nH         10^22    100.0      +/-      0.        1
    2    2    2   highecut   cutoffE  keV      10.00      +/-      0.        1
    3    3    2   highecut   foldE    keV      15.00      +/-      0.        1
    4    4    3   powerlaw   PhoIndex          1.000      +/-      0.        1
    5    5    3   powerlaw   norm              1.000      +/-      0.        1
    6    6    4   phabs      nH         10^22    1.000      +/-      0.        2
    7    2    5   highecut   cutoffE  keV      10.00      = par   2          2
    8    3    5   highecut   foldE    keV      15.00      = par   3          2
    9    4    6   powerlaw   PhoIndex          1.000      = par   4          2
   10    5    6   powerlaw   norm              1.000      = par   5          2
```

```
 ------------------------------------------------------------------------
 ------------------------------------------------------------------------
Chi-Squared =     2.0263934E+07 using    66 PHA bins.
Reduced chi-squared =      337732.2     for     60 degrees of freedom
Null hypothesis probability =    0.
```

Notice how the summary of the model, displayed immediately above, is different now that we have two groups, as opposed to one (as in all the previous examples). We can see that of the 10 model parameters, 6 are free (i.e., 4 of the second group parameters are tied to their equivalents in the first group). Fitting this model results in a huge χ^2 (not shown here), because our assumption that only a change in absorption can account for the spectral variation before and after eclipse is clearly wrong. Perhaps scattering also plays a role in reducing the flux before eclipse. This could be modeled (simply at first) by allowing the normalization of the power law to be smaller before eclipse than after eclipse. To decouple tied parameters, we change the parameter value in the second group to a value—any value—different from that in the first group (changing the value in the first group has the effect of changing both without decoupling). As usual, the `newpar` command is used:

```
XSPEC>newpar 10 1
    7 variable fit parameters
 Chi-Squared =     2.0263934E+07 using    66 PHA bins.
 Reduced chi-squared =      343456.5     for     59 degrees of freedom
 Null hypothesis probability =    0.
XSPEC>fit
...
  ------------------------------------------------------------------------
  Model:  phabs[1]*highecut[2]( powerlaw[3] )
  Model Fit Model Component  Parameter  Unit     Value                   Data
  par   par comp                                                         group
    1    1    1   phabs      nH       10^22      20.23     +/-  0.1823      1
    2    2    2   highecut   cutoffE  keV        14.68     +/-  0.5552E-01   1
    3    3    2   highecut   foldE    keV        7.430     +/-  0.8945E-01   1
    4    4    3   powerlaw   PhoIndex            1.187     +/-  0.6505E-02   1
    5    5    3   powerlaw   norm              5.8958E-02 +/-  0.9334E-03   1
    6    6    4   phabs      nH       10^22      1.270     +/-  0.3762E-01   2
    7    2    5   highecut   cutoffE  keV        14.68     = par   2         2
    8    3    5   highecut   foldE    keV        7.430     = par   3         2
    9    4    6   powerlaw   PhoIndex            1.187     = par   4         2
   10    7    6   powerlaw   norm              0.3123     +/-  0.4513E-02   2
  ------------------------------------------------------------------------
  ------------------------------------------------------------------------
 Chi-Squared =      15424.73     using    66 PHA bins.
 Reduced chi-squared =      261.4362     for     59 degrees of freedom
 Null hypothesis probability =    0.
```

After fitting, this decoupling reduces χ^2 by a factor of six to 15,478, but this is still too high. Indeed, this simple attempt to account for the spectral variability in terms of “blanket” cold absorption and scattering does not work. More sophisticated models, involving additional components and partial absorption, should be investigated.

4.5 Using XSPEC to Simulate Data: an Example from ASCA

In several cases, analyzing simulated data is a powerful tool to demonstrate feasibility. For example:

- *To support an observing proposal.* That is, to demonstrate what constraints a proposed observation would yield.
- *To support a hardware proposal.* If a response matrix is generated, it can be used to demonstrate what kind of science could be done with a new instrument.
- *To support a theoretical paper.* A theorist could write a paper describing a model, and then show how these model spectra would appear when observed. This, of course, is very like the first case.

Here, we'll use XSPEC to see how an *ASCA* observation of the elliptical galaxy NGC 4472 can constrain the condition of the hot gas. The first step is to define a model on which to base the simulation. The way XSPEC creates simulated data is to take the current model, convolve it with the current response matrix, while adding noise appropriate to the integration time specified. Once created, the simulated data can be analyzed in the same way as real data to derive confidence limits.

We begin by looking in the literature for the best estimate of the NGC 4472 spectrum. *BBXRT* observed the galaxy in 1990 and the results were published in Serlemitsos et al., (1993). They found a flux in the 0.5–4.5 keV range of 6.7×10^{-12} erg cm^{-2} s^{-1}, a temperature range of $0.74 < kT < 0.98$, an abundance range (as a fraction of solar) of $0.09 < A < 0.46$ and a column range of $5.0 \times 10^{20} < N_H < 3.7 \times 10^{21}$ cm^{-2}. A Raymond-Smith spectral model was found to give a good fit. We specify this model at first with the median parameter values, except for the normalization of the Raymond-Smith, which we leave at its default value of unity at first (but adjust later):

```
XSPEC>mo pha(ray)
  Model:  phabs[1]( raymond[2] )
Input parameter value, delta, min, bot, top, and max values for ...
Current:          1     0.001         0         0     1E+05     1E+06
phabs:nH>0.21
Current:          1      0.01     0.008     0.008        64        64
raymond:kT>0.86
Current:          1    -0.001         0         0         5         5
raymond:Abundanc>0.27
Current:          0    -0.001         0         0         2         2
raymond:Redshift>/*
  ----------------------------------------------------------------------
  ----------------------------------------------------------------------
  Model:  phabs[1]( raymond[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.2100     +/-      0.
    2    2    2   raymond    kT       keV       0.8600     +/-      0.
    3    3    2   raymond    Abundanc           0.2700     frozen
    4    4    2   raymond    Redshift               0.     frozen
    5    5    2   raymond    norm                1.000     +/-      0.
  ----------------------------------------------------------------------
  ----------------------------------------------------------------------
```

We now can derive the correct normalization by using the commands `dummyrsp`, `flux` and `newpar`. That is, we'll determine the flux of the model with the normalization of unity (this requires a response matrix to cover the *BBXRT* band—we use a dummy response here). We then work out the new normalization and reset it:

```
XSPEC> dummy 0.5 4.5
XSPEC>flux 0.5 4.5
 Model flux  0.2802     photons ( 4.9626E-10 ergs)cm**-2 s**-1 (  0.500-  4.500)
XSPEC> newpar 5 0.014
    3 variable fit parameters
XSPEC>flux
 Model flux  3.9235E-03 photons ( 6.9476E-12 ergs)cm**-2 s**-1 (  0.500-  4.500)
```

Here, we have changed the value of the normalization (the fifth parameter) from 1 to $6.7 \times 10^{-12}/4.78 \times 10^{-10} = 0.014$ to give the flux observed by *BBXRT* (6.7×10^{-12} erg cm^{-2} s^{-1} in the energy range 0.5–4.5).

The simulation is initiated with the command `fakeit`. If the argument `none` is given, the user will be prompted for the name of the response matrix. If no argument is given, the current response will be used:

```
XSPEC>fakeit none
For fake data, file #   1 needs response file: s0c1g0234p40e1_512_1av0_8i
              ... and ancillary response file: none
```

There then follows a series of prompts asking the user to specify whether he or she wants counting statistics (yes!), the name of the fake data file (`ngc4472_sis.fak` in our example), and the integration time `T` (40,000 seconds – `cornorm` can be left at its default value).

```
Use counting statistics in creating fake data? (y) /
Input optional fake file prefix (max 4 chars): /
 Fake data filename (s0c1g0234p40e1_512_1av0_8i.fak) [/ to use default]: ngc4472_sis.fak
 T, cornorm  (1, 0): 40000
 Net count rate (cts/s) for file   1  0.3563    +/-  3.0221E-03
   using response (RMF) file...       s0c1g0234p40e1_512_1av0_8i.rsp
 Chi-Squared =      188.6545     using   512 PHA bins.
 Reduced chi-squared =     0.3706375     for    509 degrees of freedom
 Null hypothesis probability =  1.00
```

We now have created a file containing a simulated spectrum of NGC 4472. As is usual before fitting, we need to check which channels to ignore. This time, we'll examine the actual numbers of counts in each channel and reject those that have fewer than 20 per channel. We use `iplot counts` and see that our criterion requires us to ignore channels 1–15 and 76–512:

```
XSPEC>ignore 1-15 76-**
 Chi-Squared =      63.30437     using    60 PHA bins.
 Reduced chi-squared =      1.110603     for     57 degrees of freedom
 Null hypothesis probability = 0.264
```

As expected, χ^2 is reasonable even before fitting because the model and the data have the same shape. But the point of this simulation is to determine confidence ranges. First, we thaw the value of the abundance (fixed by default), fit and then use the `error` command:

```
XSPEC> thaw 3
 Number of variable fit parameters =    4
XSPEC>fit
 Chi-Squared    Lvl  Fit param # 1     2           3           4
                 5
```

```
  55.3176     -3     0.2309      0.8569      0.2772          0.
            1.4322E-02
  55.2946     -4     0.2320      0.8565      0.2784          0.
            1.4322E-02
  55.2945     -5     0.2321      0.8565      0.2784          0.
            1.4322E-02
 ---------------------------------------------------------------------------
 Variances and Principal axes :
              1     2     3     5
 1.51E-08 | -0.03 -0.01  0.03  1.00
 1.02E-05 |  0.39  0.91 -0.11  0.03
 9.98E-05 | -0.91  0.40  0.11 -0.03
 2.32E-04 | -0.15 -0.06 -0.99  0.03
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
  Model:  phabs[1]( raymond[2] )
  Model Fit Model Component  Parameter  Unit     Value
  par   par comp
    1    1    1   phabs      nH       10^22     0.2321     +/-  0.9426E-02
    2    2    2   raymond    kT       keV       0.8565     +/-  0.5048E-02
    3    3    2   raymond    Abundanc           0.2784     +/-  0.1510E-01
    4    4    2   raymond    Redshift               0.     frozen
    5    5    2   raymond    norm               1.4322E-02 +/-  0.5423E-03
  ---------------------------------------------------------------------------
  ---------------------------------------------------------------------------
 Chi-Squared =      55.29454     using    60 PHA bins.
 Reduced chi-squared =     0.9874024     for     56 degrees of freedom
 Null hypothesis probability = 0.502
XSPEC>err 1 2 3
 Parameter   Confidence Range (     2.706)
     1    0.217009        0.248102
     2    0.847909        0.864666
     3    0.254807        0.304992
```

These confidence ranges show that an *ASCA* observation would definitely constrain the parameters, especially the column and abundance, more tightly than the original *BBXRT* observation. Of course, whether these constraints are sufficient depends on the theories being tested. When producing and analyzing simulated data, it is crucial to keep in mind the purpose of the proposed observation, for the potential parameter space that can be covered with simulations is almost limitless.

4.6 Producing Plots: Modifying the Defaults

The final results of using XSPEC are usually one or more tables containing confidence ranges and fit statistics, and one or more plots showing the fits themselves. So far, all the plots shown have the default settings, but it is possible to edit plots to get closer to the appearance what you want.

The plotting package used by XSPEC is **PGPLOT**, which is comprised of a library of low-level tasks. At a higher level is **QDP/PLT**, the interactive program that forms the interface between the XSPEC user and **PGPLOT**. **QDP/PLT** has its own manual; it also comes with on-line help. Here, we show how to make some of the most common modifications to plots.

To initiate interactive plotting in XSPEC, use the command `iplot` instead of the usual `plot`. In this example, we'll take the simulated *ASCA* SIS spectrum of the previous section and make the following modifications to the `data` plot:

- Change the aspect ratio
- Change the labels
- Rescale the x-axis and y-axis
- Change the y-axis to be a logarithmic scale
- Thicken the lines and make the characters smaller to make the hardcopy look better
- Produce a postscript file

After the `iplot` command, the plot itself appears, followed by the `QDP/PLT` prompt:

```
XSPEC> setplot energy
XSPEC> iplot data
 PLT>
```

The first thing we'll do is change the aspect ratio of the box that contains the plot (*viewport* in QDP terminology). The viewport is defined by the coordinates of the lower left and upper right corners of the page, normalized so that the width and height of the page are unity. The labels fall outside the viewport, so if the full viewport were specified, only the plot would appear. The default box has a viewport with corners at (0.1, 0.1) and (0.9, 0.9). For our purposes, we want a viewport with corners at (0.2, 0.2) and (0.8, 0.7): with this size and shape, the hardcopy will fit nicely on the page and not have to be reduced for photocopying. To change the viewport, use the command `viewport` followed by the coordinates:

```
 PLT> viewport 0.2 0.2 0.8 0.7
```

Next we want to change two of the labels: the label at the top, which currently says only *data*, and the label that specifies the filename. This change is a straightforward one using the `label` command, which takes as arguments a location description and the text string:

```
 PLT> label top Simulated Spectrum of NGC 4472
 PLT> label file ASCA SIS
```

Other location descriptors are available, including `x` and `y` for the x-axis and y-axis, respectively. To get help on a QDP command, type `help` followed by the name of the command at the `PLT>` prompt. Note that QDP commands can be abbreviated, just like XSPEC commands. To see the results of changing the viewport and the labels, just enter the command `plot`:

```
 PLT> plot
```

The two changes we want to make next are to rescale the axes and to change the y-axis to a logarithmic scale. The commands for these changes also are straightforward: the `rescale` command takes the minimum and maximum values as its arguments, while the `log` command takes `x` or `y` as arguments:

```
 PLT> rescale x 0.4 2.5
 PLT> rescale y 0.01 1
 PLT> log y
 PLT> plot
```

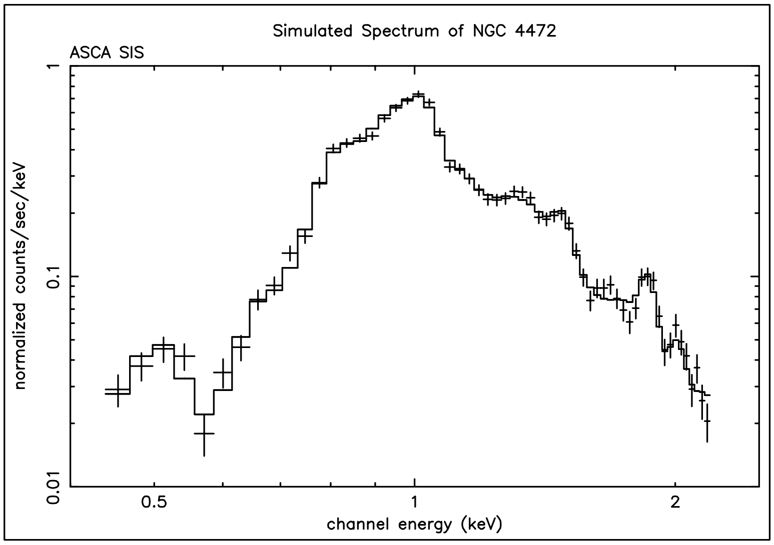

Figure G: A simulated *ASCA* SIS spectrum of NGC 4472 produced to show how a plot can be modified by the user.

To revert to a linear scale, use the command `log off y`. All that is left to change are the thickness of the lines (the default, least for postscript files that are turned into hardcopies, is too fine) and the size of the characters (we want slightly smaller characters). The `lwidth` command does the former: it takes a width as its argument: the default is 1: we'll reset it to 3. The `csize` command does the latter, taking a normalization as its argument. One (1) will not change the size, a number less than one will reduce it and a number bigger than one will increase it.

```
PLT> lwidth 3
PLT> csize 0.8
PLT> plot
```

Finally, to produce a postscript file that we can print, we use the `hardcopy` command:

```
PLT> hardcopy ngc4472_sis.ps/ps
PLT> quit
```

Here, we have given the file the name **ngc4472_sis.ps**. It will be written into the current directory. The suffix **/ps** tells the program to produce a postscript file. The `quit` command returns us to the XSPEC¿ prompt.

The result of all this manipulation is shown proudly in Figure G.

Chapter 5

XSPEC commands

5.1 Summary of Commands

abund	Set the abundance table.
addcomp	Add a component to the model.
addline	Add lines to a model (Tcl script).
arf	Read an auxiliary response file.
autosave	Periodically save the XSPEC status.
backgrnd	Reset the files to be used for background subtraction.
bayes	Set up for Bayesian inference
chatter	Control the verbosity of XSPEC.
corfile	Reset the files to be used for background correction.
cornorm	Reset the normalization to be used in correcting the background.
cosmo	Set H0, q0, and Lambda0.
cpd	Alias for `setplot device`.
data	Input one or more PHA data files.
delcomp	Delete a component from the model.
diagrsp	Diagonalize the current response for an ideal response.
dummyrsp	Create a ’dummy’ response, covering a given energy range.
dump	Write out history packages of the observed spectrum and model.
editmod	Add, delete, or replace one component in the model.
eqwidth	Calculate a model component’s equivalent width.

error Determine a single parameter confidence region.

exec Execute a shell command from within XSPEC.

exit Wind up any hardcopy plots and exit from XSPEC.

extend Extend the energy range over which the model is calculated.

fakeit Produce simulated data files for sensitivity studies.

fit Find the best fit model parameters.

flux Calculate the current model's flux over an energy range.

freeze Do not allow a model parameter to vary during the fit.

ftest Calculate the F-statistic between two model fits

gain Perform a simple modification of the response gain.

genetic Set parameters for genetic fitting method.

goodness Monte Carlo calculation of goodness-of-fit.

hardcopy Spool the current plot to the printer.

help Obtain help on XSPEC commands.

identify List possible lines in the specified energy range.

ignore Ignore a range of PHA channels in future fit operations.

improve Try to find a better minimum (requires MINUIT).

iplot As plot command but interactive using PLT.

log Open the log file to save output.

lumin Calculate the current model's luminosity over a given rest frame energy range and red-shift.

mdefine Create a new model component defined by an algebraic expression.

method Set the minimization method.

model Define the model to be used when fitting the data.

modid Guess line IDs in the model (Tcl script).

newpar Modify the model parameters.

notice Restore a range of PHA channels for future operations.

plot Plot various information on the current plot device.

query Switch on/off prompt to continue fitting.

quit An alias for `exit`.

readline	Enable/disable the gnu readline facility.
recornrm	Adjust correction norms to minimize the fit statistic, holding the model fixed.
renorm	Adjust the model norms, and/or allow automatic renorming.
response	Reset the files used to determine the detector responses.
save	Save aspects of the current state to a command file.
script	Open the script file to save all commands input.
setplot	Modify the plot device and other values used by the plot routines.
show	Display current file and model information.
source	Execute a script file.
statistic	Change the fit statistic in use.
steppar	Step through a range of parameter values; perform a fit at each step.
suggest	Make a suggestion or report a bug to the XSPEC gnomes.
syscall	Run a shell command.
systematic	Set the model systematic error.
tclout	write xspec data to tcl variable $xspec_tclout
thaw	Allow a model parameter to vary during the fit.
thleqw	Calculates expected fluorescent line equivalent width.
time	Display elapsed time and other statistical information.
uncertain	Alias for `error`.
untie	Untie linked parameters.
weight	Change the weighting function used for chi-squared fits.
xhistory	Open a history file, in order to save fit results.
xsect	Change the photoelectric absorption cross-sections in use.
xset	Modify a number of XSPEC internal switches

5.2 Description of Syntax

The individual commands are treated in alphabetical order in the following section. The novice would be well-served by reading the treatments of the `data`, `model`, `newpar`, and `fit` commands, in that order, then the other commands as needed. The write-up for each command includes a brief description of the purpose, an outline of the correct syntax, a more detailed discussion of the command assumptions and purpose, and a series of examples. Some commands have one or more subcommands that are similarly described following the command.

In the command description, the syntax uses the following conventions.

`<arg>`	!	an argument to the command
`<arg c> =:: <arg a> <arg b>`	!	defines `<arg c>` as `<arg a>` followed by `<arg b>`
`<arg>...`	!	a repeated string of arguments of the same type
`[<arg>]`	!	is an optional argument.
`{ <arg a> | <arg b> }`	!	indicates a choice between an argument of type
	!	<arg a> or <arg b>

Exceptional responses to the command prompt are :

an empty line	– Nothing performed, prompt repeated
/	– Any remaining arguments will have the values given on the last invocation of the command
`<EOF>`	– same as `quit`
	otherwise use `/*`
`?` (or anything else)	– Write a list of the commands

5.3 Command Descriptions

5.3.1 abund

Set the abundance table used in the plasma emission and photoelectric absorption models.

```
abund      <option>
```

where <option> is either `angr`, from Anders E. & Grevesse N. (1989, Geochimica et Cosmochimica Acta 53, 197), `feld`, from Feldman U. (1992, Physica Scripta 46, 202), `aneb`, from Anders E. & Ebihara (1982, Geochimica et Cosmochimica Acta 46, 2363), `grsa` from Grevesse, N. & Sauval, A.J. (1998, Space Science Reviews 85, 161), or `wilm` from Wilms, Allen & McCray (2000, ApJ 542, 914), or `file filename`, where `filename` is an ASCII file containing 18 lines with one number on each line. All abundances are number relative to H. The tables are :

Element	angr	feld	aneb	grsa	wilm
H	1.00e+0	1.00e+0	1.00e+0	1.00e+0	1.00e+0
He	9.77e-2	9.77e-2	8.01e-2	8.51e-2	9.77e-2
C	3.63e-4	3.98e-4	4.45e-4	3.31e-4	2.40e-4
N	1.12e-4	1.00e-4	9.12e-5	8.32e-5	7.59e-5
O	8.51e-4	8.51e-4	7.39e-4	6.76e-4	4.90e-4
Ne	1.23e-4	1.29e-4	1.38e-4	1.20e-4	8.71e-5
Na	2.14e-6	2.14e-6	2.10e-6	2.14e-6	1.45e-6
Mg	3.80e-5	3.80e-5	3.95e-5	3.80e-5	2.51e-5
Al	2.95e-6	2.95e-6	3.12e-6	2.95e-6	2.14e-6
Si	3.55e-5	3.55e-5	3.68e-5	3.35e-5	1.86e-5
S	1.62e-5	1.62e-5	1.89e-5	2.14e-5	1.23e-5
Cl	1.88e-7	1.88e-7	1.93e-7	3.16e-7	1.32e-7
Ar	3.63e-6	4.47e-6	3.82e-6	2.51e-6	2.57e-6
Ca	2.29e-6	2.29e-6	2.25e-6	2.29e-6	1.58e-6
Cr	4.84e-7	4.84e-7	4.93e-7	4.68e-7	3.24e-7

Fe	4.68e-5	3.24e-5	3.31e-5	3.16e-5	2.69e-5
Co	8.60e-8	8.60e-8	8.27e-8	8.32e-8	8.32e-8
Ni	1.78e-6	1.78e-6	1.81e-6	1.78e-6	1.12e-6

5.3.2 addcomp

Add a component to the current model.

```
addcomp      <comp #> <comp>
```

where `<comp #>` is the position in the model specification for the component and `<comp>` is its name. The user is prompted for parameter values for the component. Whenever it makes sense, XSPEC attempts to associate the component with the additive group of the component after it (see the `model` command for a description of additive groups). If you specify the component to be the last component in the model, then XSPEC will attempt to associate it with the overall multiplicative group. If it does not make sense to associated the component with an additive group, a new additive group is created.

Examples:

Suppose that the current model specification is `ga+po`, which using the `show` command would yield the description mo = gaussian[1] + powerlaw[2].

```
XSPEC> add 2 wab      !   gaussian[1]+wabs[2](powerlaw[3])
XSPEC> add 4 pha      !   (gaussian[1]+wabs[2](powerlaw[3]))phabs[4]
XSPEC> del 1          !   (wabs[1](powerlaw[2]))phabs[3]
XSPEC> add 2 zg       !   (wabs[1](zgauss[2]+powerlaw[3]))phabs[4]
XSPEC> del 3          !   (wabs[1](zgauss[2]))phabs[3]
XSPEC> add 4 bb/b     !   (wabs[1](zgauss[2]))phabs[3] + bbody/b[4]
```

5.3.3 addline

An auto-loaded Tcl script to add one or more lines to the current model in an optimum fashion.

```
addline      [<nlines>] [<modeltype>] [{fit|nofit}]
```

<nlines> additional lines are added one at a time. Line energies are set to that of the largest residual between the data and the model. For each line a fit is performed with the line width and normalization as the only free parameters. The default options are one line and a gaussian. The other <modeltype> that can be used is lorentz. If no third argument is given then the sigma and normalization of each line are fit. If “nofit” is specified then the fit is not performed but if “fit” is specified then all free parameters are fit.

5.3.4 arf

Read in one or more auxiliary response files (ARF). An ARF gives area versus energy and is used to modify the response matrix for a data set. The file must be in the OGIP standard format.

```
arf      [<filespec>...]
```

where `<filespec> =:: [<data set num>] <filename>...` and where `<filename>` is the name of the auxiliary response file to be used with the associated data set. `<data set num>` is the data set number for the first <`filename`> specified, <`data set num`> plus one is the data set number for the next file, and so on. If no <`data set num`> is given in the first <`filespec`> it is assumed to be 1. If no file specifications are entered, then none of the data set responses are modified. An error message is printed if the data set number is greater than the current number of data sets (as determined from the last use of the `data` command). A file name `none` indicates that no auxiliary response is to be used for that data set. No auxiliary response means that any incident spectrum will produce no counts for those particular channels. If a file is not found or cannot be opened for input, then the user is prompted for a replacement auxiliary response file. An <`EOF`> at this point is equivalent to `none`. See the `data` command for ways to completely remove the dataset from consideration.

Examples:

It is assumed that there are currently three data sets:

`XSPEC> arf a,b,c`	!	New files for the auxiliary response are given for all three files.
`XSPEC> arf 2 none`	!	No auxiliary response will be used for the second file.
`XSPEC> arf,,d.fits`	!	d.fits becomes the auxiliary response for the second file.

5.3.5 autosave

Set or disable autosave, which saves the XSPEC environment to a file periodically.

```
XSPEC> autosave <option>
```

where <`option`> is either `off` or a non-zero positive integer. If the option is `off`, then auto-saving is disable. If the option is `N`, the the XSPEC environment is saved every `N` commands. The saving of the environment is equivalent to the command `save all xautosav.xcm`, ie. both the file and model information is saved to the file xautosav.xcm in the local directory. Thus in case of an unexpected crash, the state of XSPEC before the crash can be restored with the command `source xautosav.xcm`. The default value for the auto-save option is 1.

5.3.6 backgrnd

Modify one or more of the files used in background subtraction.

```
backgrnd      [<filespec>...]
```

where <`filespec`> `=:: [`<`data set num`>`]` <`filename`>`...` and where <`filename`> is the name of the PHA file to be used for background subtraction. <`data set num`> is the data set number for the first <`filename`> specified, <`data set num`> plus one is the number for the next file, and so on. If no <`data set num`> is given in the first <`filespec`> it is assumed to be one (1). If no file specifications are entered, then none of the data set backgrounds are modified. An error message is printed if <`data set num`> is greater than the current number of data sets (as determined from the last use of the `data` command. A file name `none` indicates that no background subtraction is to be performed for that data set. If a file is not found or cannot be opened for input, then the user is prompted for a replacement background file. An <`EOF`> at this point is equivalent to using `none` as the background.

See the `data` command for ways of removing the data set from consideration. The user is also prompted for a replacement if the background file has a different number of PHA channels than the associated data set. A warning will be printed out if the background detector ID is different than that of the associated data set.

The current ignore status for channels is not affected by the `bkgrnd` command. (See the `ignore` and `notify` command).

Note that background files have the same format as the PHA files used for the `data` command. If the background file is not in the (old) SF format, then any grouping specification will be overridden by the grouping in the source spectral file so that the source and background are binned in the same way.

Examples:

There are currently three data sets...

`XSPEC> backgrnd a,b,c`	!	New files for background subtraction are given for all three data sets.
`XSPEC> backgrnd 2 none`	!	No background subtraction will be done for the second data set.
`XSPEC> backgrnd,,d`	!	`d.pha` becomes the background for the second data set.

5.3.7 bayes

Set up for Bayesian inference.

```
bayes      [<option> | <parameter>] <prior type>
           <hyperparameters>
```

If a parameter number is given as the first argument, then this command sets up the prior for the specified model parameter. The supported priors are `cons` (constant) and `exp` (exponential). The `cons` prior requires no hyperparameters and the `exp` prior requires a hyperparameter giving the exponential decay scale. The log prior for the `exp` case is taken to be $-\mathtt{param}/\mathtt{hparam} - \log \mathtt{hparam}$, where `param` is the fit parameter value and `hparam` the hyperparameter.

If the first argument to the `bayes` command is not a parameter number, then the three options allowed are `cons`, `on`, and `off`. The last two turn Bayesian inference on and off, respectively, and the first turns it on and gives all parameters a constant prior.

5.3.8 chatter

Control the verbosity of XSPEC.

```
chatter      <chatter level> <log chatter>
```

where <`chatter level`> and <`log chatter`> are integer values. The initial value for each argument is 10. Higher values will encourage XSPEC to tell the user more, lower values tell the user less, or make XSPEC "quieter." <`chatter level`> applies to the terminal output, while <`log chatter`> controls the verbosity in the log file. Currently, the maximum chattiness is 25. Values below five should be avoided, as they tend to make XSPEC far too obscure. Some commands may temporarily modify the chattiness, such as the `error` command. A chattiness of 25 will generate a lot of debug output.

Examples:

`XSPEC> chatter 10`	!	Set the terminal chattiness to 10, same as the initial value.
`XSPEC> chatter,,0`	!	Set the chattiness for the log file to very low. This setting essentially disables the log file output.
`XSPEC> chatter 5`	!	Make XSPEC very quiet.
`XSPEC> chatter 10 25`	!	Restore the terminal chattiness to the initial level, while in the log file XSPEC will tell all (particularly when new data files are read in).

5.3.9 corfile

Reset the files used for background correction.

```
corfile      [<filespec>...]
```

where <`filespec`> is the same as for the `backgrnd` command. The correction file can be associated with a data set to further adjust the count rates. It is a PHA file whose count rate is multiplied by the current associated correction norm (see the `cornorm` and `recornrm` command) and then subtracted from the input uncorrected data. The correction norm is not changed by running the corfile command. Default values for the correction file and norm are included in the data PHA file. Unlike the background file, the correction data does NOT contribute to the measurement error.

A file name of `none` is equivalent to no correction file used. If an input file can not be opened or found, an error message is printed and the user prompted for a replacement. As with the `backgrnd` command, the correction file is checked against the associated data set for number of channels, grouping status, and detector ID.

The current ignore status for channels is not affected by the `corfile` command. Note that correction files have the same format as the PHA files used by the `data` command.

Examples:

It is assumed that there are currently three data sets:

`XSPEC> corfile a,b,c`	!	New correction files are used for all three data sets.
`XSPEC> corfile 2 none`	!	No correction will be done for the second data set.
`XSPEC> corfile ,,d`	!	The 2nd file now uses `d.pha` as its correction.

5.3.10 cornorm

Reset the normalization used in correcting the background.

```
cornorm     [[<data set range>...]  [<cor norm>]]...
```

where `<data set range> =::  <first data set no.>-<last data set no.>` is a range of data sets to which the correction is to be applied and `<cor norm>` is the value to be used for the normalization. A decimal point (.) is used to distinguish a correction norm from a single data set `<data set range>`. If no correction norm is given, then the last value input is used (the initial value is one (1)). If no range is given, then the last single range input is modified. (See the `corfile` command.)

Examples:

Assume that there are four data sets, all with associated correction files already defined, either by default in their PHA file, or explicitly by using the `corfile` command.

`XSPEC> cornorm 1-4 1.`	!	The correction norm for all four is set to 1.0.
`XSPEC> cornorm 0.  1-2 0.3`	!	The correction norm for the last input range (which was 1-4) is set to 0., then files 1 and 2 are reset to 0.3.
`XSPEC> cornorm 4`	!	File 4 has the correction also set to 0.3.
`XSPEC> cornorm 1 4 -.3`	!	Files 1 and 4 are set to -.3.
`XSPEC> cornorm .7`	!	File 4 (as the last input single range) is set to 0.7.

5.3.11 cosmo

Set the cosmology used (i.e., H_0, q_0, and Λ_0).

cosmo   [$< H_0 >$ [$< q_0 >$ [$< lambda_0 >$]]]

where $< H_0 >$ is the Hubble constant in km/s/Mpc, $< q_0 >$ is the deceleration parameter, and $< lambda_0 >$ is the cosmological constant. If the cosmological constant is non-zero then at present XSPEC requires that the universe is flat. In this case the value of $< q_0 >$ will be ignored and XSPEC will assume that $\Omega_{matter} = 1 - lambda_0$. The default values are $H_0 = 50$, $q_0 = 0.5$, $\lambda_0 = 0$.

Examples:

`XSPEC> cosmo 100`	!	Set $H_0 = 100$ km/s/Mpc
`XSPEC> cosmo ,0.`	!	Set $q_0 = 0$.
`XSPEC> cosmo ,,0.7`	!	Set a flat universe with $lambda_0 = 0.7$

5.3.12 cpd

This command is an alias for `setplot device`.

```
cpd      {<PGPLOT device> | none}
```

5.3.13 data

Input one or more spectral data files.

```
data      <file spec>...
```

where `<filespec> =::  [[<data group #>:]<data set #>] <filename>...`

filename

If a particular file is not found or cannot be opened for input for some reason, then the user is prompted for a replacement file name. An <EOF> at this point is equivalent to typing `none`.

The default extension for all files is `.pha`, so all other extensions, (e.g. `.fak`) must be entered explicitly. The default directory is the current user directory when XSPEC is invoked.

When a new file is input, by default all its PHA channels are considered good channels for fitting and plotting purposes (see the `ignore` and `notice` commands), unless the file is replacing a previously-input file that had exactly the same number of total PHA channels, in which case the particular channels noticed are not modified.

If the file is an OGIP Type II PHA file containing multiple spectra, then the desired spectrum can be specified by appending {`n`} to the end of the filename, where `n` is the row number of the spectrum in the file. Alternatively, the spectrum can be specified by {`column_name=value`} where `column_name` is the name of a string column in the table.

XSPEC 11 allows ranges and wildcard characters for reading multiple spectra with a single command. Reading multiple spectra in this fashion is much more efficient (i.e. will execute much faster) than reading the same spectra separately.

Given a file pha2data.pha containing, say, 64 spectra, examples of use are:

```
XSPEC> data pha2data{14-26}
XSPEC> data pha2data{*}
```

Reading multiple ranges with a single command (i.e. `data pha2data {14-26,36-45}`) is, however, *not* supported: it is in most cases more efficient to read the entire file if multiple ranges of data within a single file are needed.

If all of the spectra in the file have the same response (RESPFILE) and auxiliary response (ANCRFILE), background (BACKFILE) and correction (CORRFILE) files, specified with the indicated FITS keywords, XSPEC will use these files. It is also possible, however, to specify separate response/auxiliary files for each spectrum in the file. This is done by replacing the RESPFILE, (ANCRFILE, BACKFILE, CORRFILE) string valued *keyword* with a string-valued *column* with a response/auxiliary/correction/background filename for each row. Consult the FTOOLS package documentation for details of how to modify the file.

The individual spectral data files are created outside of XSPEC by detector-specific software. They are organized as `XSPEC data` files, but often referred to as PHA files. The PHA file contains such information as integration time, detector effective area, and a scaling factor that estimates the expected size of the internal background. The data file also contains the names of the default files to be used for background subtraction and for the detector sensitivity versus incident photon energy (the response and arf files). A data file has the total observed counts for a number of channels and a factor for the size of any systematic error. Each channel is converted to a count rate per unit area (assumed cm^2). The default background file is used for background subtraction. An error term is calculated using Poisson statistics and any systematic error indicated in the file.

Any FITS NULL values will be converted to the value 1.e-32. This should have no practical effect because any channels with NULL values will presumably be marked as bad or otherwise ignored.

data set

Since XSPEC can fit simultaneously several sets of data (e.g., from multiple detectors), the `data` command allows multiple files to be input. Elsewhere in XSPEC, each set of data is referred to by its associated data set number. <`data set #`>, distinguishable from the list of file names because it is an integer, is the data set number for the first <`filename`> in the specification, <`data set #`>+1 is the data set number for the next file, and so on. If no <`data set #`> is given in the first <`filespec`> it is assumed to be one (1), so that the first data file input is data set 1, the next file is data set 2, and so on. A skipped-over argument indicates that the data set for that position (as input in an earlier invocation of `data`) will continue to be used. If the filename input is `none`, that data set is completely removed and any higher-number data sets are renumbered.

The `data` command determines the current total number of data sets. If the command line is NOT terminated by a slash (`/`), the total number of datasets is the largest data set number given for any files explicitly input, or the largest value of a <`data set num`> argument. If the line is terminated by a slash (`/`), then the current number of data sets is the previous total number of datsets or the number as determined from the command line, whichever is greater. The exception to this rule is that if there are NO arguments to the `data` command, then the number of data sets is unchanged. (To remove all the data files from consideration would require a command like `data 0`.)

data group

XSPEC places each data set into a data group. Each data group has its own set of parameters for the defined model. These parameters can be either independent from data group to data group, or they can be linked across data groups using the standard XSPEC syntax. At present, each data group must have the same model, but some components can have normalizations set to zero. If no data group is specified, then the default is to place all data sets in the same data group.

Examples:

`XSPEC> data a`	!	The file `a.pha` is read in as the first (and only) data set.
`XSPEC> data,,b`	!	`b.pha` becomes the second data set, the first data set is unmodified (e.g. it is still `a.pha`).
`XSPEC> data c 3 d,e,f`	!	`c.pha` replaces `a.pha` as the first data set; `d.pha`, `e.pha`, and `f.pha` provide the, third, fourth, and fifth data sets.

`XSPEC> data g/`	!	`g.pha` replaces `c.pha` as the first data set; the slash (/) indicates that the 2nd through the 5th data sets remain as before.
`XSPEC> data 2 none/`	!	The string none indicates that the 2nd data set (`b.pha`) is to be totally removed. The current total number of data sets thus becomes one less (4). The current data sets are `g.pha`, `d.pha`, `e.pha`, and `f.pha.`
`XSPEC> data h,,`	!	The current total number of data sets becomes 2, the current data sets are from `h.pha` and `d.pha`.
`XSPEC> data`	!	There is no change in the data status.
`XSPEC> data 1`	!	The number of data sets is set explicitly to one, that being from `h.pha`.
`XSPEC> data 1:1 a 2:2 b 3:3 c`	!	Read a.PHA into data group 1, `b.pha` into data group 2, and `c.pha` into data group 3.
`XSPEC> data 1:1 a 1:2 b 2:3 c`	!	Read `a.pha` and `b.pha` into data group 1, and `c.pha` into data group 2.
`XSPEC> data a{3}`	!	Read the third spectrum in the file `a.pha`.

5.3.14 delcomp

Delete components from the current model.

```
delcomp     <comp num range>
```

where `<comp num range>` is range of positions in the model specification of the components to be deleted.

Examples: Suppose that the current model specification is `wa(po+ga+ga)`.

`XSPEC> delcomp 3-4`	!	Changes the model to wa(po).
`XSPEC> delcomp 1`	!	Changes the model to po

5.3.15 diagrsp

Diagonalise the current response matrix for ideal response.

```
diagrsp
```

This command diagonalises the current response matrix. The response matrix is set so that the channel values are mapped directly into the corresponding energy ranges, based on the channel energies and energy response range of the current response matrix. This does not however change the effeciency (ie. effective area) as a function of energy stored for the current detector. Invoking this command will simulate a detector with prefect spectral resolution. If you wish to simulate a detector with prefect resolution and efficiency, use the `dummyrsp` command.

The previous response matrices can be reimplemented with the `response` command, with no arguments. Any use of the `data` and `notice` commands will replace the dummy response with a correct set of matrices.

5.3.16 dummyrsp

Create a "dummy" response, covering a given energy range.

```
dummyrsp      [<Low Energy> [<High Energy> [<# of ranges>
              [<log or linear> [<channel offset>
              [<channel width>]]]]]]
```

This command creates a dummy response matrix based on the given command line arguments, which will either temporarily supersede the current response matrix, or create a response matrix if one is not currently present. There are two main uses for this command: to do a "quick and dirty" analysis of uncalibrated data, and to examine the behaviour of the current model outside the range of the data's energy response.

In the first instance, one has a data set for which no response matrix is currently available. This command will create a diagonal response matrix with perfect efficiency. The response matrix will range in energy from `<Low Energy>` to `<High Energy>`, using `<# of ranges>` as the number of steps into which the range is logarithmicly or linearly divided. The detector channels are assigned to have widths of energy `<channel width>` (specified in keV), the lower bound of the first channel starting at an energy of `<channel offset>`. The response matrix is then set so that the channel values are mapped directly into the corresponding energy ranges. Then the data can be fit to models, etc., under conditions that assume a perfect detector response.

In the second instance, one can use this command to examine the current model outside the range of the energy response of the detector. When examining several aspects of the current model, such as plotting it or determining flux, XSPEC uses the current evaluation array. This, in turn, is defined by the current response files being used, which depend on the various detectors. For example, low energy datasets (such as those from the EXOSAT LEs) may have responses covering 0.05 to 2 keV, while non-imaging proportional counters can span the range from 1 to 30 keV. If the user wishes to examine the behavior of the model outside of the current range, then he or she temporarily must create a dummy response file that will cause the model to be evaluated from `<Low Energy>` to `<High Energy>`, using `<# of ranges>` as the number of steps into which the range is logarithmicly or linearly divided.

The initial default values for the arguments are 0.01 keV, 100 keV and 200 logarithmic energy steps. If one wishes only to set the energy response range, than the `<channel width>` argument may be omitted. In this case, or in the case where no data file has been read in, all entries of the dummy response matrix are set to zero. Under these circumstances the dummyrsp has no physically correct way of mapping the model into the data PHA channels, so the user should not try to fit–or plot–the data while the dummyrsp is active.

The previous response matrices can be reimplemented with the `response` command, with no arguments. Any use of the `data` and `notice` commands will replace the dummy response with a correct set of matrices, or with no response matrix if none was originally present.

Examples:

`XSPEC> dummyrsp`	!	Create the dummy response with the default limits, initially .01, 100, and 200 bins.
`XSPEC> dummyrsp .001 1`	!	Create a dummy response with 200 bins that cover the range from .001 to 1 keV.

`XSPEC> dummyrsp ,,,500`	!	The same range, but now with 500 bins.
`XSPEC> dummyrsp ,,,,lin`	!	The same range, but now with linearly spaced bins.
`XSPEC> dummyrsp ,,,,,0.1`	!	The same range, but now create a diagonal response matrix, with channel widths of 0.1 keV.
`XSPEC> response`	!	Restore any previous correct responses.

5.3.17 dump

Write out a history package of observed and model spectra.

```
dump      [<option>]
```

Two options are available : `ecdata` and `model`, with the same meaning that they have in the `plot` command. Plotting of `unfolded` spectra is possible with the XPLOT plotting program. A `dump ecdata` and a `dump model` write out all the necessary information into the history file.

Examples:

`XSPEC> dump`	!	The first dump command uses ecdata as default. It writes a history package containing the observed PHA spectrum and the folded model versus channel energy.
`XSPEC> dump model`	!	Write history package for the model spectrum in photons/cm^2 s keV.

5.3.18 editmod

Add, delete, or replace one component in the current model.

```
editmod     [<delimiter>] <component1> <delimiter>
            <component2> <delimiter> ...  <componentN>
            [<delimiter>]
```

where <`delimiter`> is some combination of `(`, `+`, `*`, and `)`, and <`componentJ`> is one of the models known to XSPEC. The arguments for this command should specify a new model, with the same syntax as the previous model, except for one component which may be either added, deleted, or changed to a different component type. XSPEC then compares the entered model with the current model, determines which component is to be modified (prompting the user if necessary to resolve ambiguities) and then modifies the model, prompting the user for any new parameter values which may be needed.

Examples:

`XSPEC> mo wabs(po)`	!	
`XSPEC> ed wabs(po+ga)`	!	This command will add the component gauss to model in the specified place and prompt the user for its initial parameters.

`XSPEC> mo wabs(po+zg)`	!	
`XSPEC> ed po+zg`	!	This command will delete the component wabs from the model, leaving the other components and their current parameter values unchanged.
`XSPEC> mo wabs(po+po)`	!	
`XSPEC> ed wabs(po)`	!	Here an ambiguity exists as to which component to delete. In this case XSPEC will print out the current model, showing the component number for each component, and then prompt the user for which component he wants deleted.
`XSPEC> mo wabs(po+ga)`	!	
`XSPEC> ed wabs(po+zg)`	!	The component gauss will be replaced by the component zgauss, and the user will be prompted for parameter values for the new component.

5.3.19 eqwidth

Determine the equivalent width of a model component.

```
eqwidth      [[RANGE <frac range>] <model component number>]...
             [err <number> <level> | noerr]
```

The command calculates the integrated photon flux produced by an additive model component (FLUX), the location of the peak of the photon spectrum (E), and the flux (photons per keV) at that energy of the continuum (CONTIN). The equivalent width is then defined as EW = FLUX / CONTIN in units of keV. The continuum is defined to be those other components of the selected model's additive group (see the `model` command). Thus, neither components in other additive groups nor the effect of any multiplicative components (e.g., absorption) are used in the calculation of the continuum or the equivalent width.

There are certain models with a lot of structure where, were they the continuum, it might be inappropriate to estimate the continuum flux at a single energy. The continuum model is integrated (from `E(1--<frac range>)` to `E(1+<frac range>)`. The initial value of <frac range> is 0.05 and it can changed using the `RANGE` keyword.

The err/noerr switch sets whether errors will be estimated on the equivalent width. The error algorithm is to draw parameter values from the distribution and calculate an equivalent width. `<number>` of sets of parameter values will be drawn. The resulting equivalent widths are ordered and the central `<level>` percent selected to give the error range. The parameter values distribution is assumed to be a multivariate Gaussian centered on the best-fit parameters with sigmas from the covariance matrix. This is only an approximation in the case that fit statistic space is not quadratic.

Examples:

The current model is assumed to be M1(A1+A2+A3+A4+M2(A5)), where the Mx models are multiplicative and the Ax models are additive.

`XSPEC> eqwidth 3`	!	Calculate the total flux of component A2 (the third component of the model) and find its peak energy (E). The continuum flux is found by the integral flux of A1+A3+A4, using the range of 0.95E to 1.05E to estimate the flux.
`XSPEC> eqwidth range .1`	!	As before, but now the continuum is estimated from its behavior over the range 0.9E to 1.1E.
`XSPEC> eqwidth range 0`	!	Now the continuum at the single energy range (E) will be used.
`XSPEC> eqwidth range .05 2`	!	Now the component A1 is used as the feature, and A2+A3+A4 are used for the continuum. The range has been reset to the original value.
`XSPEC> eqwidth 1`	!	Illegal, as M1 is not an additive component.
`XSPEC> eqwidth 7`	!	Illegal, as there is no other additive component with A5 in its group to be used as the continuum.

5.3.20 error

Determine the confidence region for a model parameter.

```
error      [[stopat <ntrial> <toler>] [maxi-
           mum <redchi>] [<delta fit statistic>]
           [<model param range>...]]...
```

where <model param range> =:: <first param>–<last param> determines the ranges of parameters to be examined, and `<delta fit statistic>` (distinguished from the model parameter indices by the inclusion of a period (.)), is the change in fit statistic used. Each indicated parameter is varied, within its allowed hard limits, until the value of the fit statistic, minimized by allowing all the other non-frozen parameters to vary, is equal to the last value of fit statistic determined by the `fit` command plus the indicated `<delta fit statistic>`. Note that before the `error` command is executed, the data must be fitted. The initial default values are the range 1–1 and the delta fit statistic of 2.706, equivalent to the 90% confidence region for a single interesting parameter.

The number of trials and the tolerance for determining when the critical fit statistic is reached can be modified by preceeding them with the `stopat` keyword. Initially, the values are 10 trials with a tolerance of 0.01 in fit statistic. If a new minimum is found in the course of finding the error, then the calculation is aborted and control returned to the user.

The `maximum` keyword ensures that `error` will not be run if the reduced chi-squared of the best fit exceeds `<redchi>`. The default value for `<redchi>` is 2.0.

Examples:

Assume that the current model has four model parameters.

`XSPEC> error 1-4`	!	Estimate the 90% confidence ranges for each parameter.
`XSPEC> error 9.0`	!	Estimate the confidence range for parameters 1-4 with delta fit statistic = 9.0, equivalent to the 3 sigma range.

`XSPEC> error 2.706 1 3 1. 2`	!	Estimate the 90% ranges for parameters 1 and 3, and the 1. sigma range for parameter 2.
`XSPEC> error 4`	!	Estimate the 1. sigma range for parameter 4.
`XSPEC> error stop 20,,3`	!	Estimate the 1-sigma range for parameter 3 after resetting the number of trials to 20. Note that the tolerance field had to be included (or at least skipped over).

5.3.21 exec

The command to execute a shell command.

```
exec      <shell command>
```

This command executes a shell (ie. an operating system) command, and then returns control to XSPEC after it is completed. Note that if your system is setup with the standard TCL distribution, shell commands entered at the XSPEC prompt will be executed automatically if they do not match any XSPEC or TCL command.

5.3.22 exit

The command to end the current XSPEC run.

```
exit
```

After an `exit`, the current plot files are closed. An <EOF> will have an identical result.

5.3.23 extend

Extend the energy range over which the model is calculated.

```
extend      <high | low> <energy> <no.  energies> <log | lin>
```

where high or low indicates whether the energy range is to be extended above or below that from the response matrix, `<energy>` is the maximum or minimum energy for the extension, `<no. energies>` is the number of energy bins to add, and log or lin is whether they should be spaced logarithmically or linearly. The defaults are to extend on the high end with logarithmic binning. Note that all response matrices in use are extended if necessary. This command is intended for use with convolution models which may need to have their input models calculated over a wide energy range. When the response matrix is read in again the extension is lost (note that this occurs when the `notice` command is used).

Examples:

`XSPEC> extend high 50. 50`	!	Extend the response energy to 50 keV in 50 logarithmic steps
`XSPEC> extend low 0.1 10 lin`	!	Extend the response energy down to 0.1 keV using 10 linear steps

5.3.24 fakeit

Produce PHA files with simulated data.

```
fakeit      [<file spec>...]
```

where `<file spec> =::  [<file number>] <file name>...` is similar to the syntax used in the `backgrnd`, `corfile`, and `response` command.

The `fakeit` command is used to create a number of artificial PHA files, where the current model is folded through response curves and then added to a background file. Poisson statistics can be included optionally. The integration time and correction norm are requested for each file. By default, the background, response, correction file, and numerical information are taken from the currently-defined data. If the argument line is empty, then it is assumed that the number of `fakeit` files produced is equal to the current number of data sets. The file names input as arguments are PHA files to be used as background. In these cases, their default response files and numerical data are used as the defaults when creating the associated fake data file. If `none` is given as an argument, then the user is prompted for the response file to be used and the default numerical data is set to 1., except the correction norm, which is set to zero. For each file, the user will be prompted for a PHA file name. If a background file is in use then this command will also fake a new background for each faked PHA file. Background files are given the same names as faked PHA files but with _bkg appended to the end of the root.

After the PHA files are created, they are read in automatically as the current datafiles. The ignore status is completely reset.

Examples:

There are currently four data files, the second of which has a current background file `CURRB.PHA` and the last of which has no background being subtracted, equivalent to a background file `none`.

`XSPEC> fakeit backa,,none 4`	!	Four fake data PHA files are created. The first and second use BACKA.PHA and CURB.PHA for background, and their associated response files for the response. The third and fourth files will have no background files; the user will be prompted for the response file to be used. The 4 indicates that four data files will be created. Faked background files will also be created to go with the first 2 faked data files.

5.3.25 fit

Find the best fit model parameters for the current data.

```
fit      [<no iter> [<delta fit statistic>]]
```

where `<no iter>` is the maximum no. of iterations, and `<delta fit statistic>` is the critical change in the fit statistic. The default values are 10 and 0.01, respectively. After each iteration, the value of chi-squared and the parameters are printed.

Examples:

`XSPEC> fit`	!	Fit with the default number of iterations and critical delta chi-squared.
`XSPEC> fit 60`	!	Fit with 60 as the number of iterations.
`XSPEC> fit ,,1.e-5`	!	Fit with 1.e-5 as the critical delta.

5.3.26 flux

Calculate the flux of the current model between certain limits.

```
flux      [<lowEnergy> [<hiEnergy>]] [err <number> <level> |
          noerr]
```

where `<lowEnergy>` and `<hiEnergy>` are the values over which the flux is calculated. Initial default values are 2 to 10 keV. The flux is given in units of photons/cm^2/s and ergs/cm^2/s. The energy range must be contained by the range covered by the current data sets (which determine the range over which the model is evaluated). Values outside this range will be reset automatically to the extremes. Note that the energy values are two separate arguments, and are NOT connected by a dash. (see parameter ranges in the freeze command). The err/noerr switch sets whether errors will be estimated on the flux. The error algorithm is to draw parameter values from the distribution and calculate a flux. `<number>` of sets of parameter values will be drawn. The resulting fluxes are ordered and the central `<level>` percent selected to give the error range. The parameter values distribution is assumed to be a multivariate Gaussian centered on the best-fit parameters with sigmas from the covariance matrix. This is only an approximation in the case that fit statistic space is not quadratic.

Examples:

The current data have significant responses to data within 1.5 to 18 keV.

`XSPEC> flux`	!	Calculate the current model flux over the default range.
`XSPEC> flux 6.4 7.0`	!	Calculate the current flux over 6.4 to 7 keV.
`XSPEC> flux 1 10`	!	The flux is calculated from 1.5 keV (the lower limit of the current response's sensitivity) to 10 keV.

5.3.27 freeze

Do not allow indicated model parameters to vary. (See also `thaw`.)

```
freeze     [<param range>...]
```

where `<param range> =:: {<param #> | <param #>--<param #>}`. The indicated model parameter or range of model parameters will be marked so they cannot be varied by the `fit` command by setting their associated `<delta>` to less than zero (see `newpar` command). By default, the range will be the last range input by either a `freeze` or `thaw` command.

Examples:

Currently there are six parameters, initially all unfrozen.

`XSPEC> freeze 2`	!	Parameter 2 is frozen.
`XSPEC> freeze 4-6`	!	Parameters 4, 5, and 6 are frozen.
`XSPEC> thaw 2 3-5`	!	Parameters 2, 4, and 5 are thawed, parameter 3 is unaffected.
`XSPEC> freeze`	!	Parameters 3,4,5 are frozen (the last range input by a freeze or thaw command).

5.3.28 ftest

Calculate the F-statistic and its probability given new and old values of χ^2 and number of degrees of freedom (DOF).

```
ftest     chisq2 dof2 chisq1 dof1
```

The new χ^2 and DOF, chisq2 and dof2, should come from adding an extra model component to (or thawing a frozen parameter of) the model which gave chisq1 and dof1. If the F-test probability is low then it is reasonable to add the extra model component. WARNING : it is not correct to use the F-test statistic to test for the presence of a line (see Protassov et al astro-ph/0201457).

5.3.29 gain

Modify a response file gain, in a particularly simple way. ∗CAUTION∗ This command is to be used with extreme care for investigation of the response properties. To properly fit data, the response matrix should be recalculated explicitly (outside of XSPEC) using any modified gain information derived. The 3 variants of this command are:

```
gain     fit
```

```
gain     nofit
```

```
gain     <data set #> <slope> <intercept>
```

The initial default `<data set #>` is 1; later, the default is the number of the file last modified. Initially, all responses are assumed to have nominal gains, determined implicitly by the data in the response files. This is equivalent to a <`slope`> of 1 and an <`intercept`> of zero. If the `fit` argument is given then the <`slope`> and <`intercept`> for each dataset will be added to the model parameters and can be fit for. The `nofit` argument switches off the fitting and leaves the gain at the current values of the parameters.

Unless `gain fit` is in use the gain is automatically reset to nominal whenever a new data file or response file is defined for that file number, but not by any use of the `ignore`, `notice`, or `dummyrsp` commands.

algorithm

The `gain` command shifts the energies on which the response matrix is defined and shifts the effective area curve to match. The effective area curve stored by XSPEC is either the ARF, if one was in use, or is calculated from the RSP file as the total area in each energy range. This means that if there are sharp features in the response then these will only be handled correctly by the gain command if they are in the ARF or if no ARF is input. The new energy is calculated by `E/`<`slope`> `-` <`intercept`>.

Examples:

We assume that the current response for data set one is `rspfile`.

`XSPEC> gain 1 0.98`	!	The first data set's response is adjusted with a slope of 0.98.
`XSPEC> gain 1,,.03`	!	The offset also is moved now by 0.03 channels.
`XSPEC> gain ,, 1 0`	!	Now the response is explicitly restored to the nominal value.
`XSPEC> gain ,, 1.1 -.5`	!	Now it is adjusted, with slope 1.1 and offset -0.05.
`XSPEC> response 1 none`	!	The response is removed.
`XSPEC> response rspfile`	!	`rspfile.rsp` is used as the response, the gain is reset to nominal.

5.3.30 genetic

```
genetic      <option> <value>
```

Set parameters used in the PIKAIA genetic global minimization algorithm (see ‘method’ for further details). For details see Charbonneau, P., 1995, Ap.J. Suppl, 101, 309.

Option can take the following values.

npop	Change the size of the population.
ndigits	Change the encoding accuracy.
pcross	Change the crossover rate.
imutate	Change the size of the population.
pmutate	Change the initial mutation rate.
pmutmin	Change the minimum mutation rate.
pmutmax	Change the maximum mutation rate.
fdif	Change the fitness differential.
irepro	Change the reproduction plans.
ielite	Change the elitism.
iverbose	Change the verbosity.

5.3.31 goodness

Perform a Monte Carlo calculation of the goodness-of-fit.

```
goodness      [<# of realizations>] [sim | nosim]
```

This command simulates <# of realizations> spectra based on the model and writes out the percentage of these simulations with the fit statistic less than that for the data. If the observed spectrum was produced by the model then this number should be around 50%. This command only works if the sole source of variance in the data is counting statistics. The sim/nosim switch determines whether each simulation will use parameter values drawn from a Gaussian distribution centered on the best fit with sigma from the covariance matrix. The sim switch turns on this option, nosim turns it off in which case all simulations are drawn from the best-fit model.

5.3.32 hardcopy

Spool the current plot to the printer.

```
hardcopy
```

This command takes what ever is the currently display in you plot window, writes it to a postscript file, and then sends it to a printer using the unix `lpr` command. It will thus be printed on whatever printer `lpr` uses as your default printer.

5.3.33 help

Obtain help on the XSPEC commands, their syntax, and examples of their use.

```
help      [<topic list>]
```

The `help` command uses the XHELP interactive help facility. The optional topic list can be used to go directly to a particular topic and sub-topic. The XHELP facility has a number of special input characters used to access different sub-topics. The string `??` will produce a summary of these sub-topics. To exit back to XSPEC, use an <EOF> or type <RETURN> until you leave the help facility.

Examples:

In the following example, only the results of the initial invocation (in response to the XSPEC> prompt) are given. Subsequent XHELP characters and sub-topic names can be given, until control is returned to XSPEC via the <EOF>.

XSPEC> help	!	Go to the top of the help file.
XSPEC> help command fit	!	Go to the help text for the fit command.
XSPEC> he com he exam	!	Go to the example text for the help command.
XSPEC> help model pow	!	Go to the help text for the powerlaw model.

5.3.34 identify

List possible lines in the specified energy range.

```
identify      <energy> <delta_energy> <redshift> <line_list>
```

The energy range searched is <energy> $\pm$ <Δ_{energy}> in the rest frame of the source. <line list> specifies the list of lines to be searched. The options are bearden, which searches the Bearden compilation of fluorescence lines (Bearden, J.A., 1967, Rev.Mod.Phys. 39, 78), mekal, which uses the lines from the mekal model (q.v.) and apec, which uses the APEC (http://hea-www.harvard.edu/APEC) line list. The apec option takes an additional two arguments : the temperature of the plasma (keV) and a minimum emissivity of lines to be shown. Currently the APEC version used is 1.10. If the environment variable $XSPEC_APEC is set then the identify command uses the filename $XSPEC_APEC concatenated with _line.fits instead of the standard APEC data file.

5.3.35 ignore

Ignore data channels. (See also notice.)

```
ignore      <range1> [<range2>] ...  [<rangeN>]
```

where <rangeI> =:: {<data set range>:<channel range> | <channel range>}. If no <data set range> is given, then the previous range is used (the initial default range is file one (1) only). <data set range> =:: {<init data set> -- <last data set> | <data set>} where <init data set>, <last data set>, and <data set> are data set numbers, in the order that they were input with the data command. <channel range> =:: { <initial channel> -- <last channel> | <channel>}. If integers are given for the channel ranges then channels will be used while if reals are given then energies (or wavelenths if setplot wave has been specified). If only the last channel is indicated, then a default value of one (1) is used for the initial channel. Channels remain ignored until they are explicitly noticed with the notice command, or if a data set is replaced with another data set with a different number of PHA channels. When a channel is ignored, the counts and other arrays are compressed, freeing up memory. Thus, the energy range covered by the current response matrix may be reduced.

Examples:

Assume that 4 data sets have been read in, the first 2 with 100 channels and the last 2 with 50 channels.

XSPEC> ignore **:1-10	!	The first 10 channels of all 4 data sets are ignored.

`XSPEC> ignore 80-**`	!	An attempt will be made to ignore channels $\geq$80 in all four data sets (as that was the last data set range specified). As a result,only channels 80-100 will be ignored for data sets 1 and 2. No change will occur for data sets 3 and 4, as they have no channels greater than 50.
`XSPEC> ign 4:1-20 3:30-40 45-**`	!	Channels 11-20 for data set 4 are ignored (1-10 were ignored already) while channels 30-40 and 45-50 of data set 3 are ignored.
`XSPEC> ignore 1:1-5`	!	No channels are ignored, as these were ignored at the beginning.
`XSPEC> ign 2:1.-5.`	!	Ignore all channels between 1 and 5 keV in the second dataset.

5.3.36 improve

Try to find a new minimum.

```
improve
```

This command runs the MINUIT improve command which attempts to find a new local minimum in the vicinity of the current minimum. This gives some global minimization capability.

5.3.37 iplot

Interactive plotting on the current plot device.

```
iplot     <plot type>
```

This command works like the `plot` command (see the `plot` command description), but allows the user to change the plot and to add text to the plot interactively using the PLT package. See Appendix A and Section 4.6.

5.3.38 log

Open a log file, and control whether command lines are echoed.

```
log     <log file> <log command chat> <indirect adjustment>
```

where `<log file>` is the name of the file to be opened (default extension is `.log`). If no arguments are on the line, then the default file name is `xspec.log`. If `<log file>` matches the string `none`, then the current log file is closed. `<log command chat>` is an integer that determines the importance of echoing the input command lines on the log file. The line is not echoed if `<log command chat>` is greater than the current log chattiness level (see the `chatter` command). `<indirect adjustment>` is an integer added to the value of `<log command chat>` for every nested level of indirection in the command processor (i.e., the use of command files). The default values of `<log command chat>` and `<indirect adjustment>` are 10 and 1.

Examples:

In these examples we assume that the current log chatter value is 10 and the commands are being processed in a command file that was invoked from the terminal level, so that the nesting level is one.

`XSPEC> log`	!	Turn on the log file (default `xspec.log`). Commands will not be logged from the indirect file, but will be logged on return to the terminal level.
`XSPEC> log none`	!	Close the log file.
`XSPEC> log mylog 9`	!	Open the log file (`mylog.log`) and now even the indirect commands will be logged (at least for the first level of indirection).
`XSPEC> log ,, 12 0`	!	Now even at the terminal level the commands are not logged.
`XSPEC> chatter ,, 12`	!	Modifying the log file chattiness will restore command logging even at arbitrarily deep levels of indirection.

5.3.39 lumin

Calculate the luminosity of the current model for a given redshift and rest frame energy range.

```
lumin     [<lowEnergy> [<hiEnergy>] [<redshift>]
```

where `<low Energy>` and `<hi Energy>` are the rest frame energies over which the luminosity is calculated and `<redshift>` is the source redshift. Initial default values are 2 to 10 keV for 0 redshift. The luminosity is given in units of ergs/s. The energy range redshifted to the observed range must be contained by the range covered by the current data sets (which determine the range over which the model is evaluated). Values outside this range will be automatically reset to the extremes. Note that the energy values are two separate arguments and are NOT connected by a dash (see parameter ranges in the `freeze` command description). The err/noerr switch sets whether errors will be estimated on the luminosity. The error algorithm is to draw parameter values from the distribution and calculate a luminosity. `<number>` of sets of parameter values will be drawn. The resulting luminosities are ordered and the central `<level>` percent selected to give the error range. The parameter values distribution is assumed to be a multivariate Gaussian centered on the best-fit parameters with sigmas from the covariance matrix. This is only an approximation in the case that fit statistic space is not quadratic.

Examples:

The current data have significant response to data within 1 to 18 keV.

`XSPEC> lumin,,,0.5`	!	Calculate the current model luminosity over the default range for z=0.5
`XSPEC> lumin 6.4 7.0`	!	Calculate the current luminosity over 6.4 to 7 keV.

5.3.40 mdefine

Define a simple model using an algebraic expression.

```
mdefine      [<name> [<expression> [:  [<type>] [<emin emax>]]]
```

where <name> is the name of the model, with maximum length of 8 characters. If <name> is a previously defined model with mdefine, the current definition will overwrite the old one, and the user is warned; if it is a built-in model, however, the user will be asked to use a different name. <expression> is a string of algebraic expressions, with maximum length 512 characters. The simple rules for expressions are :

1. The energy term, or the radius term for mixing model, must be 'e' or 'E' in the expression. Other words, which are not numerical constants nor internal functions, are assumed to be model parameters.

2. If a convolution model varies with the location on the spectrum to be convolved, the special variable `.e` or `.E` may be used to refer to the convolution point.

3. The maximum number of model parameters is 10.

4. The expression may contain spaces for better readability.

`:` <type> can be used to specify the type of the model. Valid types are (add, mul, mix, con). Please note that the character ":" must be used to separate the options from the <expression>. If <type> is not given, the default is `add`. Note that if type = `mix`, then the <expression> defines a brightness profile, and the special name `e` or `E` in the expression is taken to be the radius variable.

<emin emax> specify the minimum and maximum energy values for the model. Default values are 1.e-20 and 1.e+20, respectively.

Note that `mdefine` can also be used to display and delete previously defined models. The command `mdefine` with no arguments will display the name, type, and expression of all defined models. A single argument of a model name will display name, type, and expression just for that model. The model name followed by `:` will delete the specified model.

Operators

The following operators are recognized in an expression

+	addition operator
-	substraction operator
	multiplication operator
/	division operator
*	(or ^) exponentiation operator

Functions

The following intrinsic functions are supported. `mdefine` is case-*insensitive* (e.g. EXP = exp).
Unary :

EXP (expr)	=	exponential of an expression
SIN (expr)	=	sine of vector expression in rad
SIND (expr)	=	sine of an expression in degree
COS (expr)	=	cosine of an expression in rad
COSD (expr)	=	cosine of an expression in degree
TAN (expr)	=	tangent of an expression in rad
TAND (expr)	=	tangent of an expression in degree
LOG (expr)	=	base 10 log of an expression
LN (expr)	=	natural log of an expression
SQRT (expr)	=	square root of an expression
ABS (expr)	=	absolute value of an expression
INT (expr)	=	integer part of an expression
ASIN (expr)	=	arcsin of an expression in rad
ACOS (expr)	=	arccos of an expression in rad
MEAN (expr)	=	mean value of an expression
DIM (expr)	=	dimension of an expression
SMIN (expr)	=	minimum value of an expression
SMAX (expr)	=	maximum value of an expression

Binary :

MAX (expr1, expr2)	=	maximum of the two expressions
MIN (expr1, expr2)	=	minimum of the two expressions

Examples

```
XSPEC>mdef dplaw E**p1 + f*E**p2
```

! define a model named ”dplaw” with 3 parameters, p1, p2, f

```
XSPEC>mdef junk a*e+b*log(e)/sin(e)
```

! define a model named ”junk” with 2 parameters (a, b)

```
XSPEC> mdef junk2  exp(-a*e) : mul
```

! define a multiplicative model with one parameter, a

```
XSPEC> mdef junk3 0.2+B*e :  mix
```
! define a mixing model named ”junk3” with 1 parameter, B

```
XSPEC> mdef bb E**2/T**4/(exp(E/T)-1)
```
! try to define a blackbody model with name ”bb”

```
Warning: "bb" is a pre-defined model.
Please use a different name for your model.
```

```
XSPEC> mdef sg  exp(-Eˆ2/(2*A*.E)) / sqrt(6.283*A*sqrt(.E))  :  con
```

! this defines a Gaussian convolution model with sigma varying with square root of energy.

```
                          !
 XSPEC> mdef junk2 :      !   delete junk2
 XSPEC> mdef              !   display all user-defined models
```

```
- Name - T -------- Expression -----
dplaw    1  E**p1+f*E**p2
junk     1  a*E+b*LOG(E)/SIN(E)
junk3    3  a+b*E
sg       4  EXP(-E^2/(2*A*.E))/SQRT(6.283*A*SQRT(.E))
------------------------------------
```

5.3.41 method

Set the minimization method.

```
method     <algorithm> [<# of trials/evaluations> [<critical
           delta> [<start temperature> [<temperature reduction
           factor>]]]]
```

where `<algorithm>` is the method in use and the other arguments are control values for the minimization. Their meanings are explained under the individual methods. Note that all but Lev-Marq and Anneal require the MINUIT library from CERN to be linked into XSPEC. If any of the MINUIT library methods are set then the `error` command will use the MINUIT MINOS command to find the confidence regions.

anneal

```
method anneal      [<# of eval> [<crit delta> [<start temp> [<temp
                   reduction factor>]]]]
```

An experimental global minimization method using a simulated annealing algorithm. This needs tuning to work well for X-ray data. `<# of eval>` is the number of function evaluations to perform before giving up, `<crit delta>` is the convergence criterion, `<start temp>` is the initial annealing temperature, and `<temp reduction factor>` is the fractional reduction in temperature for each cycle.

genetic

```
method genetic      [<# of generations> [<init|noinit>]]
```

A global minimization method based on evolution by natural selection. The algorithm used is described by Charbonneau, P., 1995, ApJSuppl, 101, 309. If the init flag is set then the initial population is generated randomly from the entire allowed ranges of the fit parameters. If noinit is set then the algorithm continues from the last population generated. Note that this method is slow - tests with 3 free parameters show that thousands of generations are required to converge on the correct fit.

leven

```
method leven      [<# of trials> [<crit delta>]]
```

The default XSPEC minimization method using the modified Levenberg-Marquardt based on the CURFIT routine from Bevington. `<# of trials>` is the number of trial vectors before the user is prompted to say whether they want to continue fitting. `<crit delta>` is the convergence criterion.

migrad

```
method migrad      [<# of eval> [<crit delta>]]
```

The MINUIT MIGRAD method. `<# of eval>` is the number of function evaluations to perform before giving up and `<crit delta>` is the convergence criterion.

minim

```
method minim      [<# of eval> [<crit delta>]]
```

The MINUIT MINIMIZE method, uses MIGRAD unless it gets into trouble in which case it switches to SIMPLEX. `<# of eval>` is the number of function evaluations to perform before giving up and `<crit delta>` is the convergence criterion.

monte

```
method monte      [<# of eval> [<crit delta>]]
```

The MINUIT SEEK method, a simple random sampling of parameter space (not recommended !). `<# of eval>` is the number of function evaluations to do before giving up and `<crit delta>` is the convergence criterion.

simplex

```
XSPEC> method simplex [<# of eval> [<crit delta>]]
```

The MINUIT SIMPLEX method. `<# of eval>` is the number of function evaluations to perform before giving up and `<crit delta>` is the convergence criterion.

5.3.42 model

Define the form of the theoretical model to be fit to the data.

```
model     [<delimiter>] <component1> <delimiter>
          <component2> <delimiter> ...  <componentN>
          [<delimiter>]
```

where `<delimiter>` is some combination of `(`, `+`, `*`, `)`, and `<componentJ>` is one of the models known to XSPEC. Descriptions of these models may be accessed using the `help model` command. The models are divided into four types: additive, multiplicative, convolution and mixing models. Additive models are those directly associated with sources, such as power laws, thermal models, emission lines, etc. The net effect of two independent additive models is just the sum of their individual emissivities. Multiplicative models, on the other hand, do not directly produce photons, but instead modify (by an energy-dependent multiplicative parameter) the spectrum produced by one or more additive components. Examples of multiplicative models are photoelectric absorption models, edges, absorption lines, etc. Convolution models are like multiplicative models in that they modify the spectrum, but unlike multiplicative models they modify the spectrum as a whole, allowing for more complex operations than bin by bin multiplication factors. An example of a convolution model is a gaussian smoothing with energy dependent sigma. Note that when using convolution models, ordering of components may become significant. See under "combination rules" for a explanation of the order in which convolution and multiplicative models are applied.

Mixing models are a special case for multiple datagroups and mix the model up between the datagroups.

A list of all the currently installed models is given in response to the component `?` (this will leave the current model in use). To remove the current model, type `model none`. See the commands `delcomp`, `addcomp` and `editmod` for details on how to modify the current model without having to enter a completely new model.

Combination Rules

Model components are combined in the obvious algebraic way, with `+` seperating additive models, `*` separating multiplicative models, and parentheses to show which additive models the multiplicative models act on (syntactically, convolution and mixing models are treated as multiplicative models). The `*` need not be included next to parentheses, where it is redundant. Also, if only one additive model is being modified by one or more multiplicative models, the required brackets may be replaced by a `*`. In this case the additive model must be the last component in the grouping. Thus

M1*(A1+A2) + M2*M3(A3) + M4*A4 + A5

is a valid model, where the M's signify multiplicative models and the A's additive models. One may also specify one additional nesting of parentheses, so that one or more overall multiplicative components are specified, along with the possiblity of having additive components outside the overall multiplicative grouping. Thus

M0(M1*(A1+A2) + M2*M3(A3) + A4) + A5

is a valid model.

The old style syntax for entering models (versions 9.02 and earlier) is supported in version 11, but may not be supported in future versions. The following restriction applies, however: any background components must be entered on the command line as the last components in the component list.

When XSPEC parses the model syntax, it organizes the model components into additive groups, each group containing additive components and the multiplicative components which modify them. In addition there is an overall additive group, which contains the overall multiplicative components and any additve components outside the overall multiplicative grouping. Therefore, if Aij is the ith additive model in the jth group, and similarly Mij is a multiplicative model, the total spectrum is given by

M10*M20*... * ([(A11+A21+...)*M11*M21*...] + [(A12+A22+...)*...] + ...) + A10 + A20 + ...

Each component may have one or more parameters that can be varied during the fit (see the `newpar` command writeup).

When using convolution models, the order in which they are applied in the additive group can be significant. For example, the two models

C1*M1(A1+A2)
M1*C1(A1+A2)

are not necessarily equivalent (here the C's represent convolution models). The way XSPEC handles the ordering of components is by first computing the spectrum for the additive components of a given additive group (A1+A2 in the above example). It then applies all multiplicative or convolution components in the additive group from right to left, in the order they appear in the model formula. For example, in the model

C1*M2(A1+A2)C3*M4

The multiplicative and convolution components would be applied in the order M4, C3, M2, C1. Overall multiplicative and convolution components are handled similiarly. Any mixing models in an additive group are applied last, after any convolution or multiplicative models.

Background Models

A component can be specified as a background model by appending the string `/b` to the component name. A background model is convolved with the instrumental redistribution matrix but not with the effective area. This provides a means for including a description of detector-instrumental and particle backgrounds. If a response (RMF) matrix but not an auxiliary (ARF) file has been input, then the effective-area curve is calculated by assuming that the rows of the RMF sum to one. This will lead to errors at the top and bottom of the energy range of the detector.

Examples:

Note that `po` and `ga` are additive models, and that `wabs` and `phabs` are multiplicative models.

```
XSPEC> mo po                ! The single component po (powerlaw) is the model.
XSPEC> mo po+ga
XSPEC> mo (po+ga)wabs
XSPEC> mo phabs(po+ga)
XSPEC> mo wa(phabs(po)+ga)
XSPEC> mo wa po phabs ga    ! Same model using old style syntax (deprecated).
XSPEC> mo wa*phabs*po
XSPEC> mo (po+po)phabs      ! Note that though the first and second
                            ! components are the same form, their
                            ! parameters are varied separately.
XSPEC> mo phabs*wa(po)
XSPEC> mo pha(po)+ga+po/b   ! Here po is a background model.
```

5.3.43 modid

Auto-loaded Tcl script that guesses IDs for line components in the current model.

```
modid     [{<delta> | conf}]
```

For each gaussian or lorentzian line in the model the `identify` command is run. If a number is given as an argument then that is used as the delta energy for `identify`. If the string "conf" is given as the argument then the last calculated confidence regions are searched for possible line IDs. If no argument is given then "conf" is assumed.

5.3.44 newpar

Adjust one or more of the model parameters.

```
newpar      [<index range> [<param spec list> | <coupling
            list>]]


newpar      0
```

where

```
<param spec list> =::  <param value> <delta> <param range spec>
<param range spec> =::  <hard min> <soft min> <soft max> <hard max>
<coupling list> =:: = <index> [expression]
```

The model parameters are accessed through their model parameter indices. For example, the first parameter of the first model component generally is model parameter 1, etc. The first command line argument, `<index range>`, gives the indices' parameters to be modified by the `newpar` command. The default value is the range from the previous invocation of `newpar`. The remaining arguments can be used to update the parameter specification. If the parameter specification is omitted from the command line, then the user is explicitly prompted for it.

The first two arguments of the parameter specification are:

`<param value>`	!	the trial value of the parameter used initially in the fit;
`<delta>`	!	the "step size" used in the numerical determination of the derivatives used in the fitting process, when delta is set to zero, the parameter is not adjustable during the fit.

The four arguments of the range specification determine the range of acceptable values for the parameter. The soft limits should include the range of expected parameter behavior. Between the hard and soft limits, the parameter is made stiffer to adjustment by the minimization routine invoked by the `fit` command. The parameter is never allowed to have a value at or outside the hard limits. (One exception to this rule : the parameter can equal one of the hard limits if the soft limits have been placed outside the hard limit).

A slash (`/`) will set all the six parameter specification values (value, delta, range specification) to the previous value (default for a new model, current value if the parameter has previously been set or fit). The sequence `/*` leaves all parameters unchanged.

Coupling of parameters allows two or more parameters in a model to always have the same value or to be linearly relate. The coupled parameters are nevertheless free in the fit. An equal sign ("=") after the first index indicates that coupling of parameters is desired. The coupling expression is of the form

```
x       = {*} a {/ b }(+|-) c
```

The initial '*' (indicating a following multiplicative factor) is optional if the next token is a real value.

That is, `newpar x = a` α produces a coupling such that parameter `x` is `a *` α , whereas `newpar x = a -`α produces a constant offset between x and a (even if the intention was to give a sign to α.

`newpar 0` is a special form of the command, which simply prints out the current values of all the parameters.

Examples:

The total number of model parameters for the example is four.

`XSPEC> newpar 2 0.1`	!	The value of the second parameter is set to 0.1.
`XSPEC> newpar 3`	!	The program will prompt for a specification for the 3rd parameter (comp gives the name of the corresponding model component
`comp:param3>0.001,0`	!	which has its value set to 0.001 and its delta set to zero, fixing it in later fits. The program now prompts for a specification for the 4th parameter,
`comp:param4>21`	!	which is set to 21. As there is no 5th parameter, the program displays a summary and returns to command level.
`XSPEC> newpar ,,.001`	!	The value of the delta of the 3rd parameter (which is the default index as it was the first parameter modified in the previous newpar invocation) is set to 0.001, allowing it to be adjusted during any fits.

The total number of parameters for this example is six.

`XSPEC> newpar 4 = 1`	!	The value of parameter 4 is set to the value of parameter 1. This has the consequence of model parameter 4 being frozen at the value of parameter 1 during subsequent fitting procedures. If model parameter 1 is a free parameter, then both parameters 1 and 4 change their values simultaneously in the fit procedure.
`XSPEC> newpar 4 = 1 * 4/5 + 6.7`	!	The value of parameter 4 is set to 4/5 = 0.8 times the value of parameter 1 plus 6.7
`XSPEC> newpar 6 = 3 .1 - 9.5`	!	The value of parameter 6 is set to 0.1 times the value of parameter 3 minus 9.5
`XSPEC> newpar 5 = 2 + 5.`	!	The value of parameter 5 is set to the value of parameter 2 plus 5.
`XSPEC> newpar 8 = 1 / 4.6`	!	parameter 8 is set to parameter 1 divided by 4.6
`XSPEC> untie 6`	!	Makes parameter 6 independent of parameter 3 and a free parameter.

5.3.45 notice

Notice data channels. (See also `ignore`.)

```
notice     <range1> <range2> ...  <rangeN>
```

where `<rangeI> =:: {<data set range>:<channel range> | <channel range>}`. If no `<data set range>` is given, then the previous range is used (the initial default range is file one (1) only). `<data set range> =:: {<init data set> -- <last data set> | <data set>}`, where `<init data set>`, `<last data set>`, and `<data set>` are data set numbers, in the order they were input with the `data` command. `<channel range> =:: {<initial channel> -- <last channel> | <channel>}`. If `<channel range>` are integers then channels will be used or if reals then energies (or wavelenths if `setplot wave` has been specified). If only the last channel is indicated, then a default value of 1 is used for the initial channel. Channels remain noticed until they are explicitly ignored with the `ignore` command. When a data set is replaced by another data set with a different number of PHA channels, all input channels automatically are noticed. When a channel is noticed, the response matrix must be recreated to include the new channel's response. Thus all the response files are re-read (this might take a while). The energy range covered by the current response matrix may thereby be increased.

Examples:

Assume that 4 data sets have been read in, the first 2 having 100 channels and the last 2 having 50 channels. Assume also that channels 1-10 of all four data sets are ignored and that channels 80-100 of data sets 1 and 2 are ignored.

`XSPEC> notice **:1--10`	!	The first 10 channels of all 4 data sets are noticed.
`XSPEC> notice 80--**`	!	An attempt will be made to notice channels ≥80 in all 4 data sets (as that was the last data set range specified) but the result is that only channels 80–100 will be noticed for data sets 1 and 2, with no change for data sets 3 and 4 as they have no channels greater than 50.
`XSPEC> notice 1:1--5`	!	No channels are noticed, as these channels were noticed in the beginning.

5.3.46 plot

Make one or more plots to the current plot device (see `setplot device`).

```
plot      <plot type> [<plot type>]
```

`<plot type>` is a keyword describing the various plots allowed. Any of the options `counts`, `data`, or `ldata` can be followed by a second option, which should be one of `chisq`, `delchi`, `ratio`, `residuals`, or `none`. If one of these second options is given, then a two-pane plot is drawn with the first option in the upper pane and the second option in the lower. (see also `iplot`)

chisq

Plot contributions to `chisq`. The contribution is plotted +ve or -ve depending on whether the residual is +ve or -ve.

contour

Plot a two-dimensional fit-statistic contour plot of the last steppar run.

```
XSPEC> plot contour [<min fit stat> [<# levels> [<levels>]]]
```

where <min fit stat> is the minimum fit statistic relative to which the delta fit statistic is calculated, <# levels> is the number of contour levels to use and <levels> := <level1> ... <levelN> are the contour levels in the delta fit statistic.

contour will plot the fit statistic grid calculated by the last steppar command (which should have gridded on two parameters). A small plus sign ("+") will be drawn on the plot at the parameter values corresponding to the minimum found by the most recent fit.

The fit statistic confidence contours are often drawn based on a relatively small grid (i.e., 5x5). To fully understand what these plots are telling you, it is useful to know a couple of points concerning how the software chooses the location of the contour lines.

The contour plot is drawn based only on the information contained in the sample grid. For example, if the minimum fit statistic occurs when parameter 1 equal 2.25 and you use steppar 1 1.0 5.0 4, then the grid values closest to the minimum are 2.0 and 3.0. This could mean that there are no grid points where delta-fit statistic is less than your lowest level (which defaults to 1.0). As a result, the lowest contour will not be drawn. This effect can be minimized by always selecting a steppar range that causes XSPEC to step very close to the true minima. For the above example, using steppar 1 1.25 5.25 4, would have been a better selection.

The location of a contour line between grid points is designated using a linear interpolation. Since the fit statistic surface is often quadratic, a linear interpolation will result in the lines being drawn inside the true location of the contour. The combination of this and the previous effect sometimes will result in the minimum found by the fit command lying outside the region enclosed by the lowest contour level.

Examples:

XSPEC> step 2 0.5 1. 4 3 1. 2. 4	!	Create a 5∗5 grid for parameters 2 and 3
XSPEC> plot contour	!	Plot out a grid with three contours with delta fit statistic of 2.3, 4.61 and 9.21
XSPEC> pl cont,,4,1.,2.3,4.61,9.21	!	same as above, but with a delta fit statistic = 1 contour.

counts

Plot the data (with the folded model, if defined) with the y-axis being numbers of counts in each bin.

data

Plot the data (with the folded model, if defined).

delchi

Plot the residuals in terms of sigmas with error bars of size one.

dem

Plot the relative contributions of plasma at different temperatures for multi-temperature models. This is not very clever at the moment and only plots the last model calculated.

eemodel

Plot the current incident model spectrum in $E^2 f(E)$ (Note: This is NOT the same as an "unfolded" spectrum.) or, if wavelength plotting has been set, $\lambda^2 f(\lambda)$. The contributions of the various additive components also are plotted.

eeufspec

Plot the unfolded spectrum and the model in $E^2 f(E)$ or, if wavelength plotting has been set, $\lambda^2 f(\lambda)$. The contributions to the model of the various additive components are also plotted. This corresponds to the ν–f_ν plot beloved of AGN researchers. WARNING ! This plot is not model-independent and your unfolded model points will move if the model is changed.

efficien

Plot the total response efficiency versus incident photon energy. Only currently noticed channels are included in the calculation of the efficiency.

emodel

Plot the current incident model spectrum in $Ef(E)$ (Note: This is NOT the same as an unfolded spectrum.) or, if wavelength plotting has been set, $\lambda f(\lambda)$. The contributions of the various additive components also are plotted.

eufspec

Plot the unfolded spectrum and the model in $Ef(E)$ or, if wavelength plotting has been set, $\lambda f(\lambda)$. The contributions to the model of the various additive components also are plotted. WARNING ! This plot is not model-independent and your unfolded model points will move if the model is changed.

icounts

Integrated counts and folded model. The integrated counts are normalized to one.

insensitv

Plot the insensitivity of the current data set to changes in the incident spectra (experimental).

ldata

Plot the data (with the folded model, if defined) with a logarithmic y-axis.

model

Plot the current incident model spectrum (Note: This is NOT the same as an unfolded spectrum.) The contributions of the various additive components also are plotted.

ratio

Plot the data divided by the folded model.

residuals

Plot the data minus the folded model.

sensitvty

Plot the sensitivity of the current data set to changes in the incident spectra (experimental).

sum

A pretty plot of the data and residuals against both channels and energy.

ufspec

Plot the unfolded spectrum and the model. The contributions to the model of the various additive components also are plotted. WARNING ! This plot is not model-independent and your unfolded model points will move if the model is changed.

5.3.47 query

Switch on/off the continue fitting questions.

```
query      <option>
```

where <`option`> is `yes`, `no`, or `on`. If `on` then the continue fitting question in `fit`, `steppar`, and `error` will be asked when the number of trials is exceeded. Also when the number of trials to find the `error` is exceeded a question will be asked. For either of the other two options the questions will not be asked but the answer will be assumed to be `yes` or `no` depending on the <`option`> set. So to ensure that fitting continues without any questions being asked use the command `query yes`.

5.3.48 quit

The same as `exit`

```
quit
```

5.3.49 readline

Enable/disable gnu readline facility.

```
readline      [on | off]
```

This command is used to enable or disable the gnu readline command editing/history facility. Giving the command with no arguments will print the current status (enabled or disabled). Readline is enabled by default. For more information on using readline see Appendix D.

5.3.50 recornrm

Adjusts the indicated data sets' correction norm by a single multiplicative factor that minimizes the fit statistic. This approach to fitting the correction norm is considered an INTERIM solution.

```
recornrm      [<file range>...]
```

where `<data set range> := <low data set> [-<high data set>]` All the data sets specified by the one or more data set ranges on a single invocation of the command are multiplied by a single number, which is chosen to best reduce the fit statistic. If no ranges are given, then the last SINGLE range input is used. If you wish to fit a data set's correction norm individually, then refer only to that data set.

Note : The use of the `recornrm` command requires that a response and model be defined. It is perhaps best if a preliminary fit is performed. The user then should alternate between `fit`'s and `recornrm`'s until a stable solution is achieved. If the model is not a good fit, this process may not converge.

Examples:

`XSPEC> recornrm **`	!	All the files' correction norms are adjusted by a single number.
`XSPEC> recornrm 1-3 5`	!	Files 1,2,3, and 5 are adjusted.
`XSPEC> recornrm`	!	File 5 (the last range input) is adjusted by itself.
`XSPEC> recornrm 2`	!	File 2 is adjusted

5.3.51 renorm

Renormalize model, or change `renorm` conditions.

```
renorm      [AUTO | NONE | PREFIT]
```

The `renorm` command will adjust the normalizations of the model by a single multiplication factor, which is chosen to minimize the fit statistic. Such a renorm will be performed explicitly whenever the command is used without a key-word, or during certain XSPEC commands, as determined by the following key-words:

- AUTO – Renormalize after a `model`, `newpar`, and before a `fit`.
- PREFIT – Renormalize only before a `fit`.
- NONE – Perform no automatic renormalizations, i.e., only perform them when a `renorm` command is given explicitly.

5.3.52 response

Modify one or more of the matrices used to describe the response of the associated data set to incident X-rays.

```
response      [<filespec>...]
```

where `<filespec> =:: [<data set num>] <file name>...`, and where `<file name>` is the name of the response file to be used for the response of the associated data set. `<data set num>` is the data set number for the first <file name> in the specification, <data set num>+1 is the number for the next file, and so on. If no <data set num> is given in the first <filespec>, it is assumed to be one (1). If no file specifications are entered, then none of the data set responses are modified. An error message is printed if the data set number is greater than the current number of data sets (as determined from the last use of the data command. A file name `none` indicates that no response is to be used for that data set. This situation means that any incident spectrum will produce no counts for those particular channels. If a file is not found or can not be opened for input, then the user is prompted for a replacement response file. An <EOF> at this point is equivalent to using `none` as the response. See the `data` command for ways to totally remove the data set from consideration. The user is also prompted for a replacement if the response file has a different number of PHA channels than the associated data set. A warning will be printed out if the response detector ID is different from the associated data set's.

The current ignore status for channels is not affected by the command. (See the `ignore` and `notice` commands).

Examples:

It is assumed that there are currently three data sets:

`XSPEC> response a,b,c`	!	New files for the response are given for all three files.
`XSPEC> response 2 none`	!	No response will be used for the second file.
`XSPEC> response,,d`	!	`d.pha` becomes the response for the second file.

5.3.53 save

Save aspects of the current state to a command file.

```
save      <option> <filename>
```

If no <file name> is given, then the file `savexspec.xcm` is created. If you don't give the extension to the file name the default is `.xcm`. The values of <option> allowed are `model`, `files` and `all`. The `model` option writes out commands to recreate the current model and parameter values, the `files` option writes out commands to read-in the current data sets, and the `all` option does both of the above. The default option is `model`. To recover the saved context use the command `source filename`.

Examples:

`XSPEC> save,,fname`	!	Write out model commands to the file `fname.xcm`.
`XSPEC> save`	!	Same as above into file `savexspec.xcm`.
`XSPEC> save files fname`	!	Write out data file commands.

5.3.54 script

Open a script file.

```
script      <script file>
```

where <`script file`> is the name of the file to be opened (default extension is `.xcm`). If no arguments are on the line, then the default file name is `xspec.xcm`. If <`script file`> matches the string `none`, then the current script file is closed. The script file saves all commands that are input. This command is useful for users who use the same set of commands repeatedly. Once a script file is written and saved, the user then can re-run the same set of commands on other data by `souce <script file>`.

Examples:

`XSPEC> script`	!	Turn on the script file (default `xspec.xcm`).
`XSPEC> script none`	!	Close the script file.
`XSPEC> script myscript`	!	Open the script file (`myscript.xcm`).

5.3.55 setplot

Set one of the various plot options.

```
XSPEC> setplot <subcommand string>
```

where <`subcommand string`> is a keyword followed in some cases by arguments.

add

Show individual additive model components on the data plots.

```
XSPEC> setplot add
```

The opposite is `setplot noadd`.

channel

Change the x-axis on data and residual plots to channels.

```
XSPEC> setplot channel
```

command

Add a PLT command to the command list.

```
XSPEC> setplot command <PLT command>
```

where `<PLT command>` is any valid PLT command. Every time you use `setplot command`, that command is added to the list that is passed to PLT when you use `plot` or `iplot`. The most common use of `setplot command` is to add a common label to all plots produced. You should be careful when using this command, because XSPEC does not check to see if you have entered a valid PLT command. These commands are appended to the list that XSPEC creates to generate the plot and so `setplot command` will override these values (this can either be a bug or a feature, depending on what you have done!)

See also `setplot delete` and `setplot list`.

Example:

```
XSPEC> setp c LA OT Crab     !    Add the label "Crab" to future plots.
```

delete

Delete a PLT command from the command list.

```
XSPEC> setplot delete <command #>
```

where `<command #>` is the number of a PLT command that had been entered previously using `setplot command`. This command is used to delete commands from the list passed to PLT when you use the XSPEC `plot` or `iplot` commands.

Example:

```
XSPEC> setp c LA OT Testing    !   PLT label command
XSPEC> setp c LWidth 5         !   PLT line-width command
XSPEC> setplot lis             !   List the PLT command stack.
1:  LAbel OT Testing           !
2:  LWidth 5                   !
XSPEC> setplot del 1           !   Delete the first command in the stack.
XSPEC> setplot lis             !
1:  LWidth 5                   !
```

device

Set the device used for plots.

```
XSPEC> setplot device {<PGPLOT device> | NONE}
```

where `<PGPLOT device>` is a string that gives an (optional) file name and device plot type (see below and the PGPLOT documentation for options available). Entering the device `none` will close the current plot device and disable plotting until a subsequent `setplot device` is issued. Note that if an environment (UNIX) or logical (VMS) variable called `PGPLOT_TYPE` exists, then XSPEC automatically sets it as the plot device on invocation.

PGPLOT devices A number of plot device types are supported in XSPEC. PGPLOT devices available on Unix machines are :

```
/GIF        Graphics Interchange Format file, landscape orientation
/VGIF       Graphics Interchange Format file, portrait orientation
/NULL       Null device, no output
/PPM        Portable Pixel Map file, landscape orientation
/VPPM       Portable Pixel Map file, portrait orientation
/PS         PostScript file, landscape orientation
/VPS        PostScript file, portrait orientation
/CPS        Colour PostScript file, landscape orientation
/VCPS       Colour PostScript file, portrait orientation
/TEK4010    Tektronix 4010 terminal
/GF         GraphOn Tek terminal emulator
/RETRO      Retrographics VT640 Tek emulator
/GTERM      Color gterm terminal emulator
/XTERM      XTERM Tek terminal emulator
/ZSTEM      ZSTEM Tek terminal emulator
/V603       Visual 603 terminal
/KRM3       Kermit 3 IBM-PC terminal emulator
/TK4100     Tektronix 4100 terminals
/VT125      DEC VT125 and other REGIS terminals
/XDISP      pgdisp or figdisp server
/XWINDOW    X window window@node:display.screen/xw
/XSERVE     A /XWINDOW window that persists for re-use
```

Examples:

```
XSPEC> setplot device /xt      !   sets the device to the xterm.
XSPEC> setplot device none     !   closes the plot file.
```

energy

Change the x-axis on plots to energy in units of keV.

```
XSPEC> setplot energy
```

group

Define a range of data sets to be in the same group for plotting purposes only.

```
XSPEC> setplot group <data set range>...
```

where <data set range> is a range of contiguous data sets to be treated as a single data set for plotting purposes. The data sets still are fit individually. If multiple ranges are given, each range becomes a single group. Initially, all data sets read in are treated as single data sets.

Examples:

Assume that there are five data sets currently read in, all of them ungrouped initially.

`XSPEC> setplot group 1-4`	!	The first four data sets are treated as one group, with the fifth data sets on its own. Thus all plots will appear to have two data sets.
`XSPEC> setplot group 1 2 3 4`	!	The data sets are reset to each be in their own group.
`XSPEC> setplot group 2-3 4-5`	!	Now there are three plot groups, being data set 1, by itself, and data sets 2-3 and 4-5 as groups.
`XSPEC> setplot group 1-**`	!	All the data sets are placed in a single plot group.

id

Switch on plotting of line IDs.

```
XSPEC> setplot id <temperature> <emissivity limit> <redshift>
```

The IDs are taken from the APEC line list for the temperature given by the first argument. The plot only shows those lines with emissivities above the limit set and the lines are redshifted by the amount specified. Currently the APEC version used is 1.10. If `xset apecroot` has been used to reset the APEC files then setplot id uses as the filename the rootname concatenated with _line.fits instead of the standard APEC data filename.

list

List all the PLT commands in the command list.

```
XSPEC> setplot list
```

See `setplot delete` for an example of use.

noadd

Do not show individual additive model components on the data plots.

```
XSPEC> setplot noadd
```

The opposite is `setplot add`.

noid

Switch off plotting of line IDs.

rebin

Define characteristics used in rebinning the data (for plotting purposes ONLY).

```
XSPEC> setplot rebin <min significance> <max # bins> <plot
group> <error type>
```

In plotting the data from a data set (or group of data sets, see `setplot group`), adjacent bins are combined until they have a significant detection at least as large as `<min significance>` (in sigma). However, no more than `<max # bins>` may be so combined. Initial values are 0. and 1, respectively. This argument effects only the presentation of the data in plots. It does not change the fitting, in particular the number of degrees of freedom. The values given are applied to all the plotted data in the plot group specified as the final argument. To simultaneously change the rebinning for all the plot groups give a negative value of the plot group.

The `<error type>` argument specifies how to calculate the error bars on the new bins. The default is `quad` which sums in quadrature the errors on the original bins. `sqrt` uses $\sqrt{N}$, where N is the number of counts in the new bin, `poiss-1` uses $1+\sqrt{N+0.75}$, `poiss-2` uses $\sqrt{N-0.25}$, and `poiss-3` is the arithmetic mean of `poiss-1` and `poiss-2`. If background is present its error is calculated by the same method then added in quadrature to the source error.

Examples:

`XSPEC> setplot rebin 3 5 1`	!	Bins in plot group 1 are plotted that have at least 3 sigma, or are grouped in sets of 5 bins.
`XSPEC> setplot rebin 5 5`	!	The significance is increased to 5 sigma.
`XSPEC> setplot rebin,,10,-1`	!	All plotted bins can be grouped into up to 10 bins in reaching the 5 sigma significance criterion.
`XSPEC> setplot rebin,,,,sqrt`	!	Uses $\sqrt{N}$ to calculate error bars.

wave

Change the x-axis on plots to wavelength in units of angstroms.

```
XSPEC> setplot wave
```

This command also makes `ignore` and `notice` operate in terms of wavelength rather than energies.

5.3.56 show

List selected information to the user's terminal (and the log file, if open).

```
show     [<selection>]
```

where `<selection>` is a key word to select the information to be printed. If omitted, it is the information last asked for. Initially, the default selection is `all`. Selections are:

`all`	!	All the information.
`allfile`	!	All file information = files+noticed+rates.
`control`	!	XSPEC control information.
`free`	!	Free parameters.
`files`	!	File names, associated coefficients.
`fit`	!	Fit information.
`model`	!	The model specification.
`noticed`	!	Channel ranges noticed for each file.
`parameters`	!	Current model parameter values.
`parameters <n>`	!	Current settings for the <n>th model parameter.
`pha`	!	Current data, error and model values for each channel.
`rates`	!	Folded model, correction rates for each file.

Note that `show pha` requires the chatter level to be set to ≥ 11.

5.3.57 statistic

Change the fit statistic in use.

```
statistic      [chi | cstat]
```

The options are chi-squared (chi) or C statistic (cstat). Note that the C statistic has a couple of limitations : a) if the calculated model goes to zero in any channel then the format statistic goes to infinity - in practice this is trapped and a large value of the statistic should result; b) the C statistic can be used to estimate parameter values and confidence regions but does not provide a goodness-of-fit. See Arnaud (2001, ApJ submitted) for further details.

5.3.58 source

Add tcl procedures in script file to command set. This command is intended to be used for developing new procedures, after which the script should be added to the user's $XSPEC_HOME directory and be source'd automatically on startup.

```
source      <file name>
```

The full file name must be specified. There are no default extensions as in previous versions of XSPEC. The script may use the full power of the TCL scripting language. It is recommended that full command names be used when writing scripts. This will cause the script files to run more efficiently.

N.B. The alternative syntax for executing files containing xspec commands,

```
XSPEC>@<scriptname>
```

is recommended for xspec scripts not containing tcl procedures, for example output from the 'save' command. This is because the tcl command processor uses a 'newline' character to determine when a command that is not in curly parentheses is ended. However, this is not correct for xspec's multiple-line commands (i.e. those which when run interactively prompt the user for input such as `model`). Hence scripts that contain such commands may not process correctly. In contrast, the '@' command will detect such conditions and process the script properly.

5.3.59 steppar

Perform a fit while stepping the value of a parameter through a given range. Useful for determining confidence ranges in situations where greater control is needed than given with the `error` command.

```
steppar      <step spec> [<step spec>...]
```

where `<step spec> ::= [{log | nolog}] [{current | best}] <param index> <low value> <high value> <# steps>`. The parameter is stepped from `<low value>` to `<high value>` in `<# steps>` plus one trials. The stepping is either linear or log. Initially, the stepping is linear but it can be changed by the optional string `log` before the parameter index. `nolog` will force the stepping to be returned to the linear form. The number of steps is set initially to 10. At each value, the parameter is frozen, a fit performed, and the resulting value of chi-squared given. If `best` is given as an argument then the non-stepped parameters are reset to the best-fit values at each grid point. Alternatively, if `current` is given as an argument then the non-stepped parameters are started at their values after the last grid point (the default).

If multiple `<step spec>` are given for different parameters, then a raster scan of the parameter ranges is performed. At the end of the set, the parameters and chi-squared are restored to the values they had initially.

Examples:

Assume that the current model has four parameters:

`XSPEC> steppar 3 1.5 2.5`	!	Step parameter 3 from 1.5 to 2.5 in steps of .1.
`XSPEC> steppar log`	!	Repeat the above, only use multiplicative steps of 1.0524.
`XSPEC> step nolog 2 -.2 .2 20`	!	Step parameter 2 linearly from -.2 to .2 in steps of 0.02.

5.3.60 suggest

Get advice on what to do with bug reports and enhancement requests.

```
suggest
```

If the suggestion is the result of a catastrophic error, please give as much information as possible. Clarity and completeness will improve the response to your suggestion.

5.3.61 syscall

Execute command in a shell.

```
syscall      [<shell command>]
```

This command executes its arguments by passing them to the users current shell for execution. Thus file name globbing (ie. wildcard expansion) are performed on the command before execution. This is in contrast to the `exec` command, which executes commands directly, without first passing them on to a shell.

If no arguments are given, then the command will start an interactive subshell.

5.3.62 systematic

```
systematic     [<model systematic error>]
```

Set a systematic error term on the model to be added in quadrature to that on the data when evaluating chi-squared. The default value is zero.

5.3.63 tclout

Write internal xspec data to a tcl variable. This facility allows the manipulation of xspec data by tcl scripts, so that one can, for example, extract data from xspec runs and store in output files, format xspec output data as desired, use independent plotting software, etc.

```
tclout     <option> [<par1>] [<par2>] [<par3>]
```

`tclout` creates the tcl variable `$xspec_tclout`, which can then of course be set to any named variable. The allowed values of <option> are :

?	the valid options. Does not set $xspec_tclout.
chatter	current xspec chatter level.
compinfo n	name, 1st parameter number and number of parameters of model component n.
datagrp	number of datagroups.
datasets	number of datasets.
dof	degrees of freedom in fit.
energies n	energy array for dataset n.
eqwidth n	last equivalent width calculated for datagroup n.
error n	last last confidence region calculated for datagroup n.
filename n	filename corresponding to dataset n.
flux n	last model (photon flux,energy flux) calculated for datagroup n.
genpop n	the genetic algorithm population for parameter n.
idline e d	possible line IDs within the range [e-d, e+d].
model	description of current model.
modcomp	number of model components.
modpar	number of model parameters.
modval n	calculated model values for dataset n (in ph/cm^2/s/bin).
noticed n	range (low,high) of noticed channels for dataset n.
param n	(value,delta,min,low,high,max) for model parameter n.
peakrsid n {lo, hi}	energies and strengths of the peak residuals (+ve and -ve) for the dataset n. Optional arguments lo, hi specify an energy range in which to search.
pinfo n	parameter name and unit for parameter n.
plink	information on parameter linking for parameter n. This is in the form true/false (T or F) for linked/not linked, followed by the multiplicative factor and additive constants if linked.
plot option array n	arrays from a plot command for plot group n. The array options are x,xerr,y,yerr.
plotgrp	number of plot groups.

rate	count rate, uncertainty and the model rate for the specified dataset number.
solab	solar abundance table values.
stat	value of statistic.
varpar	number of variable fit parameters.

Examples:

```
XSPEC>data file1
XSPEC> model pha(po)
...
XSPEC> fit
...
XSPEC>tclout stat
XSPEC>scan $xspec_tclout "%f" chistat
XSPEC>tclout param 1
XSPEC>scan $xspec_tclout "%f"  par2
XSPEC>tclout param 2
XSPEC>scan $xspec_tclout "%f"  par3
XSPEC>tclout param 3
```

In this example, `scan` is a tcl command that does a formatted read of the variable `$xspec_tclout`. It reads the first floating point number ("%f") into the variable given by the last argument on the line. This sequence creates a simple model, fits it, and then writes the χ^2 statistic and the three parameters to tcl variables `$chistat`, `$par1`, `$par2` and `$par3`. These can now be manipulated in any way permitted by tcl.

5.3.64 thaw

Allow indicated parameters to vary. (See also `freeze`.)

```
thaw      [<param range>...]
```

where `<param range> =:: {<param #> | <param #>--<param #>}`. The indicated parameter, or range of parameters, will be marked as variable by the fitting commands, by setting their associated <delta> to greater than zero (see `newpar` command). (Note that parameters with <delta> values equal to zero are not affected. The `newpar` command must be used to change these.) By default, the range will be the last range input by either a `freeze` or `thaw` command. See the `freeze examples` for an example of the use of the `thaw` command.

5.3.65 thleqw

Determine the expected line equivalent width of a fluorescent line.

```
thleqw      [<line energy> <line fluorescent yield> <absorption
            edge energy> <upper energy limit> <bin number> <log
            or lin> <max edge depth>]
```

The `thleqw` command integrates the photon flux absorbed in an edge with the maximum depth from the edge energy to the upper energy limit. To get good resolution in the edge, the command uses the `dummyrsp` command with the edge energy as low energy, the upper energy limit as high energy and the bin number for the number of ranges (see `dummyrsp` command). The calculation for the edge absorption is done in the same way as in the `edge` command. The fluorescent yield is used to calculate the photon flux going into a fluorescent line emission. The continuum flux is calculated in the same way as it is in the `eqwidth` command. No lines are added to the continuum.

This model assumes a spherically symmetric distribution of absorbing material around the X-ray source with a column density obtained by the `wabs` model. The defaults are 1.0 keV, 0., .01 keV, 100. keV, 200, and 0.0, which gives 0 for the equivalent width. They are chosen this way since the `dummyrsp` command defaults are 0.01 keV, 100. keV and 200.

Since the `thleqw` command uses `dummyrsp`, the old response must be read in again when working further on the same spectrum.

Examples:

`XSPEC> thl 6.4,.34,7.1,20.,,,,.015`	!	Calculate the expected equivalent width for a line at 6.4 keV with a fluorescent yield of 34%. The absorption edge is integrated from 7.1 keV to 20. keV in 200 steps. The maximum depth is .015.
`XSPEC> thleqw ,,,,,,100`	!	As before, but with 100 steps
`XSPEC> thleqw`	!	As before.

Don't forget to read the response file in again if you want to do further work on the same data.

5.3.66 time

Get some information about the program run time.

```
time
```

Provides the following:

elapsed real time
elapsed CPU time
buffered I/O count
direct I/O count
page fault count

5.3.67 uncertain

A synonym for `error`.

5.3.68 untie

Untie the specified parameter from any links to other parameters.

```
untie      <param #>
```

5.3.69 weight

Change the weighting function used in the calculation of chi-squared.

```
weight      [standard | gehrels | churazov | model ]
```

Standard weighting uses $\sqrt{N}$ or the statistical error given in the input spectrum. Gehrels weighting uses $1 + \sqrt{N + 0.75}$, a better approximation when N is small (Gehrels, N. 1986, ApJ 303, 336). Churazov weighting uses the suggestion of Churazov etal (1996, ApJ 471, 673) to estimate the weight for a given channel by averaging the counts in surrounding channels. Model weighting uses the value of the model, not the data, to estimate the weight.

5.3.70 xhistory

Open a file for outputting history records. [deprecated]

```
xhistory      {<filename> | none}
```

where <filename> is the name of the file to be opened for output. The string `none` will close the current history file. The history file produces SF format packages that can be read by other programs. For example, the results of the XSPEC `steppar` command are placed in the `steppar` package, which in return is used by the XPLOT program to plot confidence contours. The history file is not directly human-readable. A more printable recounting of an XSPEC session is produced with the log file (see the `log` command). Currently implemented history packages:

data files	produced by	`data`, `fakeit`
assoc. files		`data`, `backgrnd`, `response`, `corfile`,`fakeit`
channels		`data`,`ignore`,`notice`
model		`addcomp`, `model`, `delcomp`
param def		`addcomp`, `model`, `define`, `delcomp`, `newpar`, `addcomp`, `model`, `freeze`
fit		`fit`
steppar		`steppar`
error		`error`
equiv width		`eqwidth`
flux		`flux`
flum		`lumin`
spectrum		`dump ecdata`
photon		`dump model`
time		`time`

5.3.71 xsect

Change the photoelectric absorption cross-sections in use.

```
xsect      [bcmc|obcm|vern]
```

The three options are `bcmc`, from Balucinska-Church & McCammon (1992; Ap.J. 400, 699) with a new

He cross-section based on (1998; Ap.J. 496, 1044); `obcm`, as `bcmc` but with the old He cross-section, and, `vern`, from Verner et. al. (1996 Ap.J.). This changes the cross-sections in use for all absorption models with the exception of `wabs`.

5.3.72 xset

Modify a number of XSPEC internal switches.

```
xset      [ abund | cosmo | mdatadir | method | statistic
          | weight | xsect | <string_name> ] [ <options> |
          <string_value> ]
```

The arguments abund, cosmo, method, statistic, weight, and xsect just run the appropriate XSPEC commands. mdatadir changes the directory in which XSPEC searches for model data files. You probably don't want to change this. The `<string_name>` option can be used to pass string values to models. XSPEC maintains a database of `<string_name>`, `<string_value>` pairs created using this command. Individual model functions can then access this database. Note that xset does no checking on whether the `<string_name>` is used by any model so spelling errors will not be trapped.
To access the `<string_name>`, `<string_value>` database from within a model function use the fortran function fgmstr. This is defined as character*128 and takes a single argument, the string name as a character*128. If the `<string_name>` has not been set then a blank string will be returned.

Examples:

`XSPEC> xset neivers 2.0`	!	Set the NEIVERS variable to 2.0
`XSPEC> xset`	!	List the current string variables

Chapter 6

XSPEC V11.2 Models

6.1 Summary of Models

absori	Ionized absorber.
ascac	ASCA PSF mixing model.
apec	APEC thermal plasma model.
atable	Additive table model.
bbody	Blackbody spectrum.
bbodyrad	Blackbody spectrum with norm proportional to surface area.
bexrav	E-folded broken power-law reflected from neutral matter
bexriv	E-folded broken power-law reflected from ionized matter
bknpower	Broken powerlaw.
bmc	Comptonization by relativistically moving matter.
bremss	Thermal bremsstrahlung.
c6mekl	6th-order Chebyshev polynomial DEM using mekal.
c6pmekl	Exponential of 6th-order Chebyshev polyn. DEM using mekal.
c6pvmkl	Variable abundance version of c6pmekl
c6vmekl	Variable abundance version of c6mekl
cabs	Compton scattering (non-relativistic)
cemekl	Multi-temperature mekal.
cevmkl	Multi-temperature vmeka.
cflow	Cooling flow model.
compbb	Comptonized blackbody spectrum after Nishimura et al. 1986.

compls Comptonization spectrum after Lamb and Sanford 1979.

compst Comptonization spectrum after Sunyaev and Titarchuk 1980.

comptt Comptonization spectrum after Titarchuk 1994.

constant Energy-independent multiplicative factor.

cutoffpl Powerlaw with high energy exponential rolloff.

cyclabs Cyclotron absorption line.

disk Disk model.

diskbb Multiple blackbody disk model.

diskline Line emission from relativistic accretion disk.

diskm Disk model with gas pressure viscosity.

disko Modified blackbody disk model.

diskpn Accretion disk around a black hole.

dust Dust scattering out of the beam.

edge Absorption edge.

equil Equilibrium ionization collisional plasma model from Borkowski.

etable Table model for exponential of -1 times the input.

expabs Low-energy exponential rolloff.

expfac Exponential factor.

gaussian Simple gaussian line profile.

gnei Generalized single ionization NEI plasma model.

grad GR accretion disk around a black hole.

grbm Gamma-ray burst model.

gsmooth Gaussian smoothing with an energy dependent sigma.

highecut High energy cutoff.

hrefl Simple reflection model good up to 15 keV.

laor Line from accretion disk around a black hole.

lorentz Lorentzian line profile.

lsmooth Lorentzian smoothing with an energy dependent sigma.

meka Mewe-Gronenschild-Kaastra thermal plasma (1992).

mekal	Mewe-Kaastra-Liedahl thermal plasma (1995).
mkcflow	Cooling flow model based on mekal.
mtable	Multiplicative table model.
nei	Simple nonequilibrium ionization plasma model.
notch	Notch line absorption.
npshock	Plane-parallel shock with ion and electron temperatures.
nteea	Pair plasma model.
pcfabs	Partial covering fraction absorption.
pegpwrlw	Powerlaw with pegged normalization.
pexrav	Exponentially cut-off power-law reflected from neutral matter.
pexriv	Exponentially cut-off power-law reflected from ionized matter.
phabs	Photo-electric absorption
pileup	CCD pile-up model
plabs	Absorption model with power-law dependence on energy.
plcabs	Cut-off powerlaw observed through dense, cold matter.
posm	Positronium continuum.
powerlaw	Simple photon power law.
projct	3-D to 2-D projection mixing model.
pshock	Constant temperature, plane-parallel shock plasma model.
raymond	Raymond-Smith thermal plasma.
redden	IR/optical/UV extinction from Cardelli et al. (1989)
redge	Recombination edge.
reflect	Convolution model for reflection from neutral matter
refsch	E-folded power-law reflected from an ionized relativistic disk.
rgsxsrc	Convolution model for extended sources with the XMM RGS.
sedov	Sedov model with electron and ion temperatures.
smedge	Smoothed absorption edge.
spline	Spline multiplicative factor.
srcut	Synchrotron radiation from cut-off electron distribution.

sresc Synchrotron radiation from escape-limited electron distribution.

SSS_ice Einstein Observatory SSS ice absorption.

step Step function convolved with gaussian.

tbabs Absorption due to the ISM including molecules and grains.

tbgrain ISM absorption with variable molecule and grain fractions.

tbvarabs ISM absorption with variable abundances and grain depletion.

uvred UV reddening.

vapec APEC thermal plasma model with variable abundances.

varabs Photoelectric absorption with variable abundances.

vbremss Thermal bremsstrahlung spectrum with variable H/He.

vequil As equil but with variable abundances.

vgnei As gnei but with variable abundances.

vmcflow Cooling flow model based on vmekal.

vmeka M-G-K thermal plasma with variable abundances.

vmekal M-K-L thermal plasma with variable abundances.

vnei As nei but with variable abundances.

vnpshock As npshock but with variable abundances.

vphabs Photoelectric absorption with variable abundances.

vpshock As pshock but with variable abundances.

vraymond Raymond-Smith thermal plasma with variable abundances.

vsedov As sedov but with variable abundances.

wabs Photoelectric absorption (Morrison & McCammon).

wndabs Photoelectric absorption with low energy window.

xion The reflected spectrum from a photo-ionized accretion disk.

zbbody Redshifted blackbody.

zbremss Redshifted thermal bremsstrahlung.

zedge Redshifted absorption edge.

zgauss Redshifted gaussian.

zhighect Redshifted high energy cut-off.

zpcfabs Redshifted partial covering absorption.

zphabs Redshifted photoelectric absorption

zpowerlw Redshifted powerlaw.

ztbabs Redshifted ISM absorption without grains.

zvarabs Redshifted photoelectric absorption with variable abundances.

zvfeabs Redshifted absorption with variable iron abundance.

zvphabs Redshifted photoelectric absorption with variable abundances.

zwabs Redshifted "Wisconsin absorption."

zwndabs Redshifted photoelectric absorption with low energy window.

6.2 Additive Model Components (Sources)

This and the following sections contain information on specific, installed XSPEC models. The parameters are given as `par1`, `par2`, ... and `K`, which is the normalization. Additive models represent sources of emission.

6.2.1 apec

An emission spectrum from collisionally-ionized diffuse gas calculated using the APEC code v1.10. If the environment variable XSPEC_APEC is set then the model uses the filename XSPEC_APEC concatenated with _coco.fits and _line.fits instead of the standard APEC data files. More information can be found at http://hea-www.harvard.edu/APEC/ which should be consulted by anyone running this model.

`par1`	=	plasma temperature in keV
`par2`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Al, Si, S, Ar, Ca, Fe, Ni. Relative abundances are set by the abund command.
`par3`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2))\int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.2 atable

An additive table model. The filename to be used should be given immediately after `atable` in the `model` command. For example

```
XSPEC> model atable{mymod.mod}
```

uses `mymod.mod` as the input for the model. For specifications of the table model file, see the OGIP memo 92-009 on the FITS file format for table model files (available on the WWW or by anonymous ftp from `ftp://legacy.gsfc.nasa.gov/caldb/docs/memos`. Example additive table model files are mekal.mod and raysmith.mod in $XANADU/spectral/xspec/manager and testpo.mod in $XANADU/src/spectral/session.

6.2.3 bbody

A blackbody spectrum.

$$A(E) = \texttt{K}\, 8.0525 E^2 dE / (\texttt{par1}^4 (\exp(E/\texttt{par1}) - 1))$$

where :

`par1`	=	temperature kT in keV
`K`	=	L_{39}/ D_{10}^2, where L_{39} is the source luminosity in units of 10^{39} ergs/sec and D_{10} is the distance to the source in units of 10 kpc

6.2.4 bbodyrad

A blackbody spectrum with normalization proportional to the surface area.

$$A(E) = \texttt{K}\, 1.0344 \times 10^{-3} E^2 dE / (\exp(E/\texttt{par1}) - 1)$$

where :

`par1`	=	temperature kT in keV
`K`	=	R_{km}^2/D_{10}^2, where R_{km} is the source radius in km and D_{10} is the distance to the source in units of 10 kpc

6.2.5 bexrav

A broken power-law spectrum multiplied by exponential high-energy cutoff, $\exp[-E\ \texttt{par4}]$, and reflected from neutral material. See Magdziarz & Zdziarski 1995, MNRAS, 273, 837 for details. The output spectrum is the sum of an e-folded broken power law and the reflection component. The reflection component alone can be obtained for $rel_{refl} < 0$. Then the actual reflection normalization is $|rel_{refl}|$. Note that you need to change then the limits of rel_{refl} excluding zero (as then the direct component appears). If $E_c = 0$, there is no cutoff in the power law. The metal and iron abundance are variable with respect to those set by a command 'abund'. The opacities are of Balucinska & McCammon (1992, and 1994, private communication). As expected in AGNs, H and He are assumed to be fully ionized. Send questions or comments to aaz@camk.edu.pl.

`par1`	=	Gamma1, first power law photon index
`par2`	=	E_{break}, break energy (keV)
`par3`	=	Gamma2, second power law photon index
`par4`	=	E_c, the e-folding energy in keV (if $E_c = 0$ there is no cutoff)
`par5`	=	rel_{refl}, reflection scaling factor (1 for isotropic source above disk)
`par6`	=	redshift, z
`par7`	=	abundance of elements heavier than He relative to the solar abundances
`par8`	=	iron abundance relative to the above
`par9`	=	cosine of inclination angle
`K`	=	photon flux at 1 keV of the cutoff broken power-law only (no reflection) in the observed frame.

6.2.6 bexriv

Broken power-law spectrum multiplied by exponential high-energy cutoff, $\exp[-E/\mathtt{par4}]$, and reflected from ionized material. See Magdziarz & Zdziarski 1995, MNRAS, 273, 837 for details. Ionization and opacities of the reflecting medium is computed as in the procedure absori. The output spectrum is the sum of an e-folded broken power law and the reflection component. The reflection component alone can be obtained for $rel_{refl} < 0$. Then the actual reflection normalization is $|rel_{refl}|$. Note that you need to change then the limits of rel_{refl} excluding zero (as then the direct component appears). If $E_c = 0$, there is no cutoff in the power law. The metal and iron abundances are variable with respect to those set by a command 'abund'. Send questions or comments to aaz@camk.edu.pl.

`par1`	=	Gamma1, first power law photon index
`par2`	=	E_{break}, break energy (keV)
`par3`	=	Gamma2, second power law photon index
`par4`	=	E_c, the e-folding energy in keV (if $E_c = 0$ there is no cutoff)
`par5`	=	rel_{refl}, reflection scaling factor (1 for isotropic source above disk)
`par6`	=	redshift, z
`par7`	=	abundance of elements heavier than He relative to the solar abundances
`par8`	=	iron abundance relative to the above
`par9`	=	cosine of inclination angle
`par10`	=	disk temperature in K
`par11`	=	disk ionization parameter, $\xi = 4\pi F_{ion}/n$, where F_{ion} is the 5 eV – 20 keV irradiating flux, n is the density of the reflector; see Done et al., 1992, ApJ, 395, 275
`K`	=	photon flux at 1 keV of the cutoff broken power-law only (no reflection) in the observed frame.

6.2.7 bknpower

A broken power law.

$$\begin{aligned} A(E) &= \mathtt{K}\,(E/1\mathrm{keV})^{-\mathtt{par1}} & \text{for } E \le \mathtt{par2} \\ &= \mathtt{K}\,\mathtt{par2}^{(\mathtt{par3}-\mathtt{par1})} \times (E/1\mathrm{keV})^{-\mathtt{par3}} & \text{for } E \ge \mathtt{par2} \end{aligned}$$

where :

`par1`	=	power law photon index for E $<$ break energy
`par2`	=	break point for the energy in keV
`par3`	=	power law photon index for E $>$ break energy
`K`	=	photons/keV/cm^2/s at 1 keV

6.2.8 bmc

This is an analytic model describing Comptonization of soft photons by matter undergoing relativistic bulk-motion. The typical scenario involves thermal X-rays from the inner region of an accretion disk in a black-hole binary illuminating in-falling matter in close proximity to the black-hole event horizon. For a detailed description of the model, refer to Titarchuk, Mastichiadis & Kylafis 1997, ApJ, 487, 834; Titarchuk & Zannias, 1998, ApJ, 493, 863; Laurent & Titarchuk 1999, ApJ, 511, 289; Zannias, Borozdin, Revnivtsev., Trudolyubov, Shrader, & Titarchuk, 1999, ApJ, 517, 367; or Shrader & Titarchuk 1999, ApJ 521, L21. The model parameters are the characteristic black-body temperature of the soft photon source, a spectral (energy) index, and an illumination parameter characterizing the fractional illumination of the bulk-motion flow by the thermal photon source. It must be emphasized that this model is not an additive combination of power law and thermal sources, rather it represents a self-consistent convolution. The bulk-motion up-scattering and Compton recoil combine to produce the hard spectral tail, which combined with the thermal source results in the canonical high-soft-state spectrum of black hole accretion. The position of the sharp high energy cutoff (due to recoil) can be determined using the theta function $\Theta(E_c - E)$. The model can also be used for the general Comptonization case when the energy range is limited from above by the plasma temperature (see compTT and compST).

`par1`	=	Temperature of thermal photon source in keV.
`par2`	=	Energy spectral index alpha.
`par3`	=	Log of the A parameter. Note that f in Borozdin et al 1999 and Shrader & Titarchuk 1999 is $10^{par(3)}$.
`K`	=	A_N defined in in Borozdin et al 1999 and Shrader & Titarchuk (1999)

6.2.9 bremss

A thermal bremsstrahlung spectrum based on the Kellogg, Baldwin & Koch (ApJ 199, 299) polynomial fits to the Karzas & Latter (ApJSuppl 6, 167) numerical values. A routine from Kurucz (priv. comm.) is used in at low temperature end. The He abundance is assumed to be 8.5% of H by number.

`par1`	=	plasma temperature in keV
`K`	=	$(3.02 \times 10^{-15}/(4\pi D^2) \int n_e n_I dV$, where D is the distance to the source (cm) and n_e, n_I are the electron and ion densities (cm^{-3})

6.2.10 c6mekl

A multi-temperature mekal model using sixth-order Chebyshev polynomial for the differential emission measure. The DEM is not constrained to be positive. The abundance is relative to the Solar abundances set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate. The reference for this model is Singh et al.(1996, ApJ, 456, 766).

`par1-6`	=	Chebyshev polynomial coefficients
`par7`	=	H density (cm^{-3})
`par8`	=	abundance wrt to Solar
`par9`	=	Redshift
`par10`	=	0 $\Rightarrow$ calculate, 1 $\Rightarrow$ interpolate
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e is the electron density (cm^{-3}), and n_H is the hydrogen density (cm^{-3})

6.2.11 c6pmekl

A multi-temperature mekal model using the exponential of a sixth-order Chebyshev polynomial for the differential emission measure. (see e.g. Lemen et al. ApJ 341, 474, 1989). The abundance is relative to the Solar abundances set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate. The reference for this model is Singh et al.(1996, ApJ, 456, 766).

`par1-6`	=	Chebyshev polynomial coefficients
`par7`	=	H density (cm^{-3})
`par8`	=	abundance wrt to Solar
`par9`	=	Redshift
`par10`	=	$0 \Rightarrow$ calculate, $1 \Rightarrow$ interpolate
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e is the electron density (cm^{-3}), and n_H is the hydrogen density (cm^{-3})

6.2.12 c6pvmkl

A multi-temperature, variable abundance mekal model using the exponential of a sixth order Chebyshev polynomial for the differential emission measure (eg Lemen etal. ApJ 341, 474, 1989). The abundances are relative to the Solar abundances set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate. The reference for this model is Singh et al.(1996, ApJ, 456, 766).

`par1-6`	=	Chebyshev polynomial coefficients
`par7`	=	H density (cm^{-3})
`par8 - par21`	=	Abundances of He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par22`	=	Redshift
`par23`	=	$0 \Rightarrow$ calculate, $1 \Rightarrow$ interpolate
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e is the electron density (cm^{-3}), and n_H is the hydrogen density (cm^{-3})

6.2.13 c6vmekl

A multi-temperature, variable-abundance mekal model using the sixth-order Chebyshev polynomial for the differential emission measure. The DEM is not constrained to be positive. The abundances are relative to the Solar abundances set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate. The reference for this model is Singh et al.(1996, ApJ, 456, 766).

`par1-6`	=	Chebyshev polynomial coefficients
`par7`	=	H density (cm^{-3})
`par8-par21`	=	Abundances of He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par22`	=	Redshift
`par23`	=	$0 \Rightarrow$ calculate, $1 \Rightarrow$ interpolate
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e is the electron density (cm^{-3}), and n_H is the hydrogen density (cm^{-3})

6.2.14 cemekl

A multi-temperature plasma emission model built from the mekal code. Emission measures follow a power-law in temperature (ie emission measure from temperature T is proportional to $(T/\mathtt{par2})^{\mathtt{par1}}$. The abundance ratios are set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate.

`par1`	=	index for power-law emissivity function
`par2`	=	maximum temperature
`par3`	=	nH (cm^{-3})
`par4`	=	Abundance relative Solar
`par5`	=	Redshift
`par6`	=	0 $\Rightarrow$ calculate, 1 $\Rightarrow$ interpolate
`K`	=	Normalization

6.2.15 cevmkl

A multi-temperature plasma emission model built from the mekal code. Emission measures follow a power-law in temperature (ie emission measure from temperature T is proportional to $(T/\mathtt{par2})^{\mathtt{par1}}$. The abundances are relative to the Solar abundances set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate.

`par1`	=	index for power-law emissivity function
`par2`	=	maximum temperature
`par3`	=	nH (cm^{-3})
`par4-par17`	=	Abundances for He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par18`	=	Redshift
`par19`	=	0 $\Rightarrow$ calculate, 1 $\Rightarrow$ interpolate
`K`	=	Normalization

6.2.16 cflow

A cooling flow model after Mushotzky & Szymkowiak ("Cooling Flows in Clusters and Galaxies," ed. Fabian, 1988). An index of zero for the power-law emissivity function corresponds to emission measure weighted by the inverse of the bolometric luminosity at that temperature. The model assumes $H_0 = 50$ and $q_0 = 0$. The abundance ratios are set by the abund command.

`par1`	=	index for power-law emissivity function
`par2`	=	low temperature (keV)
`par3`	=	high temperature (keV)
`par4`	=	abundance relative Solar
`par5`	=	redshift
`K`	=	Mass accretion rate (solar mass/yr)

6.2.17 compbb

A comptonized blackbody model after Nishimura et al., 1986, PASJ, 38, 819.

`par1`	=	blackbody temperature (keV)
`par2`	=	electron temperature of the hot plasma (keV)
`par3`	=	optical depth of the plasma
`K`	=	L_{39}^2/D_{10}^2, where L_{39} is the source luminosity in units of 10^39 ergs/sec and D_{10} is the distance to the source in units of 10 kpc (the same definition as for the bbodyrad model)

6.2.18 compLS

A Comptonization spectrum after Lamb and Sanford, 1979, *M.N.R.A.S*, 288, 555. This model calculates the self-Comptonization of a bremsstrahlung emission from an optically-thick spherical plasma cloud with a given optical depth and temperature. It was popular for Sco X-1.

`par1`	=	temperature in keV
`par2`	=	optical depth
`K`	=	Normalization

6.2.19 compST

A Comptonization spectrum after Sunyaev and Titarchuk 1980, *A&A*, 86, 121. This model is the Comptonization of cool photons on hot electrons.

`par1`	=	temperature in keV
`par2`	=	optical depth
`K`	=	$(Nf)/(4\pi d^2)$, where N is the total number of photons from the source, d is the distance to the source, and f is the factor $z(z+3)y^z/\Gamma(2z+4)/\Gamma(z)$, where z is the spectral index, y is the injected photon energy in units of the temperature, and Γ is the incomplete gamma function.

6.2.20 compTT

This is an analytic model describing Comptonization of soft photons in a hot plasma, developed by L. Titarchuk (see ApJ, 434, 313). This replaces the Sunyaev-Titarchuk Comptonization model in the sense that the theory is extended to include relativistic effects. Also, the approximations used in the model work well for both the optically thin and thick regimes. The Comptonized spectrum is determined completely by the plasma temperature and the so-called β parameter which is independent of geometry. The optical depth is then determined as a function of β for a given geometry. Thus `par5` switches between spherical and disk geometries so that β is not a direct input here. This parameter MUST be frozen. If `par5` > 0, β is obtained from the optical depth using analytic approximation (e.g. Titarchuk 1994). If `par5` < 0 and $0.1 < \tau < 10$, β is obtained by interpolation from a set of accurately calculated pairs of β and τ from Sunyaev & Titarchuk 1985 (A&A 143, 374).

In this incarnation of the model, the soft photon input spectrum is a Wien law [$x^2\ e^{-x}$ photons] because this lends itself to a particularly simple analytical form of the model. For present X-ray detectors this should be adequate. Note that in energy flux space the peak of the Wien law occurs at 3kT as opposed to 2.8kT for a blackbody.

The plasma temperature may range from 2-500 keV, but the model is not valid for simultaneously low temperatures and low optical depth, or for high temperatures and high optical depth. The user is strongly urged to read the following references (esp. HT95 Fig 7) before and after using this model in order to fully understand and appreciate the physical assumptions made: Titarchuk, L., 1994, ApJ, 434, 313; Hua, X-M., Titarchuk, L., 1995, ApJ, 449, 188; Titarchuk, L., Lyubarskij, Y., 1995, ApJ, 450, 876.

`par1`	=	Redshift.
`par2`	=	Input soft photon (Wien) temperature (keV).
`par3`	=	Plasma temperature (keV).
`par4`	=	Plasma optical depth.
`par5`	=	Geometry switch. ABS(`par5`) $<= 1$: disk, > 1: sphere. `par5` $>= 0$: use analytic approx for β vs. τ. `par5` < 0 : Get beta vs. tau from interpolation.
`K`	=	Normalization

6.2.21 cutoffpl

A power law with high energy exponential rolloff.

$$A(E) = \texttt{K}(E/1\mathrm{keV})^{-\texttt{par1}} \exp(-E/\texttt{par2})$$

where :

`par1`	=	power law photon index
`par2`	=	e-folding energy of exponential rolloff (in keV)
`K`	=	photons/keV/cm^2/s at 1 keV

6.2.22 disk

The spectrum from an accretion disk, where the opacities are dominated by free-free absorption, i.e., the so-called blackbody disk model. Not correct for a disk around a neutron star.

`par1`	=	accretion rate in Eddington Luminosities
`par2`	=	central mass in solar mass units
`par3`	=	inner disk radius in gravitational (= 3 Schwarzschild) radii
`par4`	=	distance in kpc
`K`	=	$2\cos i/d^2$, where i is the inclination of the disk and d is the distance in units of 10 kpc

6.2.23 diskbb

The spectrum from an accretion disk consisting of multiple blackbody components. For example, see Mitsuda et al., *PASJ*, 36, 741, (1984), Makishima et al., *ApJ* 308, 635, (1986).

`par1`	=	temperature at inner disk radius (keV)
`K`	=	((Rin/km)/(D/10kpc))^2 $\cos\theta$, where Rin is the inner disk radius, D the distance to the source, and θ the angle of the disk

6.2.24 diskline

A line emission from a relativistic accretion disk. See Fabian et al., *MNRAS* 238, 729. Setting `par2` to 10 is the special case of the accretion disk emissivity law ($(1-\sqrt{6/R})/R^3$).

`par1`	=	line energy
`par2`	=	power law depend. of emissivity. If this parameter is 10 or greater then the accretion disk emissivity law $(1-\sqrt{6/R})/R^3$ is used. Otherwise the emissivity scales as $R^{\texttt{par2}}$.
`par3`	=	inner radius (units of GM/c^2)
`par4`	=	outer radius (units of GM/c^2)
`par5`	=	inclination (degrees)
`K`	=	photons/cm^2/s in the spectrum

6.2.25 diskm

A disk model with gas pressure viscosity. The spectrum from an accretion disk where the viscosity scales as the gas pressure. From Stella and Rosner 1984, *ApJ*, 277, 312.

`par1`	=	accretion rate in Eddington luminosities
`par2`	=	central mass in units of solar mass
`par3`	=	inner disk radius in gravitational (= 3 Schwarzschild) radii
`par4`	=	viscosity
`K`	=	cos(i)/d^2, where i is the inclination of the disk and d is the distance in units of 10 kpc

6.2.26 disko

A modified blackbody disk model. The spectrum from the inner region of an accretion disk where the viscosity is dominated by radiation pressure.

`par1`	=	accretion rate in Eddington luminosities
`par2`	=	central mass in units of solar mass
`par3`	=	inner disk radius in gravitational (= 3 Schwarzschild) radii
`par4`	=	viscosity
`K`	=	cos(i)/d^2, where i is the inclination of the disk and d is the distance in units of 10 kpc

6.2.27 diskpn

Blackbody spectrum of an accretion disk. This is an extention of diskbb model, including corrections for temperature distribution near the black hole. The temperature distribution was calculated in Paczynski-Wiita pseudo-Newtonian potential. An accretion rate can be computed from the maximum temperature found. For details see Gierlinski et al., 1999, MNRAS, 309, 496. Please note that the inner disk radius (`par2`) can be a free parameter only close to $\texttt{par2} = 6$; otherwise par2 is strongly correlated with K.

`par1`	=	maximum temperature in the disk (keV)

`par2`	=	inner disk radius in $R_g = GM/c^2$ units, $6 <= par2 <= 1000$
`K`	=	$(M^2 \cos i)/(D^2 \beta^4)$ - normalization, where M - central mass in solar masses, D - distance to the source (kpc), i - inclination of the disk, β - color/effective temperature ratio.

6.2.28 equil

Ionization equilibrium collisional plasma model. This is the equilibrium version of Kazik Borkowski's NEI models. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature (keV)
`par2`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni. Abundances are given by the Anders & Grevesse mixture.
`par3`	=	redshift z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

The references for this model are as follows :

- Borkowski, Lyerly & Reynolds, 2001, ApJ, 548, 820
- Hamilton, A.J.S., Sarazin, C. L. & Chevalier, R. A. , 1983, ApJS, 51,115
- Borkowski, K.J., Sarazin, C.L. & Blondin, J.M. 1994, ApJ, 429, 710
- Liedahl, D.A., Osterheld, A.L. & Goldstein, W.H. 1995, ApJ, 438, L115

6.2.29 gaussian

A simple gaussian line profile. If the width is ≤ 0, then it is treated as a delta function.

$$A(E) = \mathtt{K}\ (1./\mathtt{par2}\sqrt{(2\pi)})\exp(-0.5((E - \mathtt{par1})/\mathtt{par2})^2)$$

where :

`par1`	=	line energy in keV
`par2`	=	line width (σ) in keV
`K`	=	total photons/cm^2/s in the line

6.2.30 gnei

Non-equilibrium ionization collisional plasma model. This is a generalization of the nei model where the temperature is allowed to have been different in the past ie the ionization timescale averaged temperature is not necessarily equal to the current temperature. For example, in a standard Sedov model with equal electron and ion temperatures, the ionization timescale averaged temperature is always higher than the current temperature for each fluid element. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature (keV)
`par2`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni. Abundances are given by the Anders & Grevesse mixture.
`par3`	=	Ionization timescale in units of s/cm^3.
`par4`	=	Ionization timescale averaged plasma temperature (keV).
`par5`	=	redshift z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2))\int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.31 grad

General Relativistic Accretion Disk model around a Schwarzschild black hole. Inner radius is fixed to be 3 Schwarzschild radii, thus the energy conversion efficiency is 0.057. See Hanawa, T., 1989, ApJ, 341, 948 and Ebisawa, K. Mitsuda, K. and Hanawa, T. 1991, ApJ, 367, 213. Several bugs were found in the old GRAD model which was included in xspec 11.0.1ae and before. Due to these bugs, it turned out that the mass obtained by fitting the old GRAD model to the observation was 1.4 times over-estimated. These bugs were fixed, and a new parameter (par6) was added to make the distinction between the old and new codes clear.

`par1`	=	distance (kpc)
`par2`	=	disk inclination angle (deg; 0 for face-on)
`par3`	=	mass of the central object (solar units)
`par4`	=	mass accretion rate (10^{18} g/s)
`par5`	=	spectral hardening factor, T_{col}/T_{eff}. Should be greater than 1.0, and considered to be 1.5-1.9 for accretion disks around a stellar-mass black hole. See, e.g., Shimura and Takahara, 1995, ApJ, 445, 780
`par6`	=	A flag to switch on/off the relativistic effects (never allowed to be free). If positive, relativistic calculation; if negative or zero, Newtonian calculation (inner radius is still fixed at 3 Schwarzschild radii, and the efficiency is 1/12).
`K`	=	Should be fixed to 1.

6.2.32 grbm

A model for gamma-ray burst continuum spectra developed by D. Band, et. al., 1993 (ApJ 413, 281).

$$
\begin{array}{rcll}
A(E) & = & \texttt{K}\,(E/100.)^{\texttt{par1}}\exp(-E/\texttt{par3}) & \text{for } E < (\texttt{par1}-\texttt{par2})*\texttt{par3} \\
A(E) & = & \texttt{K}(\texttt{par1}-\texttt{par2})\texttt{par3}/100.)^{(\texttt{par1}-\texttt{par2})} & \\
 & & \exp(-(\texttt{par1}-\texttt{par2})(E/100.)^{\texttt{par2}} & \text{for } E > (\texttt{par1}-\texttt{par2})*\texttt{par3}
\end{array}
$$

where:

par1	=	first power law index
par2	=	second power law index
par3	=	characteristic energy in keV
K	=	normalization constant

6.2.33 laor

An emission line from an accretion disk around a black hole. Ari Laor's calculation including GR effects (ApJ 376, 90).

par1	=	Line energy in keV
par2	=	power law depend. of emissivity (scales as $R^{-\texttt{par2}}$)
par3	=	inner radius (units of GM/c^2)
par4	=	outer radius (units of GM/c^2)
par5	=	inclination (degrees)
K	=	photons/cm^2/s in the line

6.2.34 lorentz

A Lorentzian line profile.

$$A(E) = \texttt{K}\,(\texttt{par2}/(2\pi))/((E-\texttt{par1})^2 + (\texttt{par2}/2)^2)$$

where :

par1	=	line energy in keV
par2	=	line width (σ) in keV
K	=	total photons/cm^2/s in the line

6.2.35 meka

An emission spectrum from hot diffuse gas based on the model calculations of Mewe and Gronenschild (as amended by Kaastra). The model includes line emissions from several elements.

par1	=	plasma temperature in keV
par2	=	hydrogen density in cm^{-3}
par3	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni. Abundances are set by the abund command.

`par4`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

The references for the MEKA model are as follows :

- Mewe, R., Gronenschild, E.H.B.M., and van den Oord, G.H.J. 1985, *A&AS*, 62, 197
- Mewe, R., Lemen, J.R., and van den Oord, G.H.J. 1986, *A&AS*, 65, 511
- Kaastra, J.S. 1992, An X-Ray Spectral Code for Optically Thin Plasmas (Internal SRON-Leiden Report, updated version 2.0)

Similar credit may also be given for the adopted ionization balance

- Arnaud, M., and Rothenflug, M. 1985, *A&AS*, 60, 425
- Arnaud, M., and Raymond, J. 1992, *ApJ*, 398, 394

6.2.36 mekal

An emission spectrum from hot diffuse gas based on the model calculations of Mewe and Kaastra with Fe L calculations by Liedahl. The model includes line emissions from several elements. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate.

`par1`	=	plasma temperature in keV
`par2`	=	hydrogen density in cm^{-3}
`par3`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni. Abundances are set by the abund command.
`par4`	=	redshift, z
`par5`	=	$0 \Rightarrow$ calculate, $1 \Rightarrow$ interpolate
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

The references for the MEKAL model are as follows :

- Mewe, R., Gronenschild, E.H.B.M., and van den Oord, G.H.J. 1985, *A&AS*, 62, 197
- Mewe, R., Lemen, J.R., and van den Oord, G.H.J. 1986, *A&AS*, 65, 511
- Kaastra, J.S. 1992, An X-Ray Spectral Code for Optically Thin Plasmas (Internal SRON-Leiden Report, updated version 2.0)
- Liedahl, D.A., Osterheld, A.L., and Goldstein, W.H. 1995, ApJL, 438, 115

Similar credit may also be given for the adopted ionization balance

- Arnaud, M., and Rothenflug, M. 1985, *A&AS*, 60, 425
- Arnaud, M., and Raymond, J. 1992, *ApJ*, 398, 394

6.2.37 mkcflow

A cooling flow model after Mushotzky & Szymkowiak ("Cooling Flows in Clusters and Galaxies" ed. Fabian, 1988). This one uses the mekal model for the individual temperature components and differs from `cflow` in setting the emissivity function to be the inverse of the bolometric luminosity. The model assumes $H_0 = 50$ and $q_0 = 0$. Abundance ratios are set by the `abund` command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate.

`par1`	=	low temperature (keV)
`par2`	=	high temperature (keV)
`par3`	=	abundance relative to Solar
`par4`	=	redshift
`par5`	=	$0 \Rightarrow$ calculate, $1 \Rightarrow$ interpolate
`K`	=	Mass accretion rate (solar mass/yr)

6.2.38 nei

Non-equilibrium ionization collisional plasma model. This assumes a constant temperature and single ionization parameter. It provides a characterisation of the spectrum but is not a physical model. The references for this model can be found under the description of the equil model. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature (keV)
`par2`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni. Abundances are given by the Anders & Grevesse mixture.
`par3`	=	Ionization timescale in units of s/cm^3.
`par4`	=	redshift z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.39 npshock

Plane-parallel shock plasma model with separate ion and electron temperatures. This model is slow. `par1` provides a measure of the average energy per particle (ions+electrons) and is constant throughout the post-shock flow in plane shock models (Borkowski et al., 2001, ApJ, 548, 820). `par2` should always be less than `par1`. If `par2` exceeds `par1` then their interpretations are switched (ie the larger of `par1` and `par2` is always the mean temperature). Additional references can be found under the help for the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	mean shock temperature (keV)
`par2`	=	electron temperature immediately behind the shock front (keV).
`par3`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni. Abundances are given by the Anders & Grevesse mixture.
`par4`	=	Lower limit on ionization timescale (s/cm^3) to include.
`par5`	=	Upper limit on ionization timescale (s/cm^3) to include.
`par6`	=	redshift z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.40 nteea

A nonthermal pair plasma model based on that of Lightman & Zdziarski (1987, ApJ 319, 643) from Magdziarz and Zdziarski. It includes angle-dependent reflection from Magdziarz & Zdziarski (1995, MNRAS 273, 837). The abundances are set up by the command abund. Send questions or comments to aaz@camk.edu.pl.

`par1`	=	nonthermal electron compactness
`par2`	=	blackbody compactness
`par3`	=	scaling factor for reflection (1 for isotropic source above disk)
`par4`	=	blackbody temperature in eV
`par5`	=	the maximum Lorentz factor
`par6`	=	thermal compactness (0 for pure nonthermal plasma)
`par7`	=	Thomson optical depth of ionization electrons (e.g., 0)
`par8`	=	electron injection index (0 for monoenergetic injection)
`par9`	=	minimum Lorentz factor of the power law injection (not used for monoenergetic injection)
`par10`	=	minimum Lorentz factor for nonthermal reprocessing (> 1; $<=$ `par9`)
`par11`	=	radius in cm (for Coulomb/bremsstrahlung only)
`par12`	=	pair escape rate in c (0-1, see Zdziarski 1985, ApJ, 289, 514))
`par13`	=	cosine of inclination angle
`par14`	=	iron abundance relative to that defined by abund
`par15`	=	redshift
`K`	=	photon flux of the direct component (w/o reflection) at 1 keV in the observer's frame.

6.2.41 pegpwrlw

A power law with pegged normalization.

$$A(E) = \mathtt{K}(E/1\mathrm{keV})^{-\mathtt{par1}}$$

where :

`par1`	=	photon index of power law (dimensionless)
`par2`	=	lower peg energy range

`par3`	=	upper peg energy range
`K`	=	flux (in units of 10^{-12} erg/cm^2/s) over the energy `par2`–`par3` unless `par2` = `par3`, in which case it is the flux (in micro-Jy) at `par2`

6.2.42 pexrav

Exponentially cut off power law spectrum reflected from neutral material (Magdziarz & Zdziarski 1995, MNRAS, 273, 837). The output spectrum is the sum of the cut-off power law and the reflection component. The reflection component alone can be obtained for rel_refl < 0. Then the actual reflection normalization is $|$rel_refl$|$. Note that you need to change then the limits of rel_refl excluding zero (as then the direct component appears). If $E_c = 0$ there is no cutoff in the power law. The metal and iron abundance are variable with respect to those defined by the command abund. The opacities are from Balucinska & McCammon (ApJ 400, 699 and 1994, private communication). H and He are assumed to be fully ionized. Send questions or comments to aaz@camk.edu.pl.

`par1`	=	γ, power law photon index, N_E prop. to $E^{-\gamma}$
`par2`	=	E_c, the cutoff energy in keV (if $E_c = 0$ there is no cutoff; one needs to change the lower limit for that)
`par3`	=	rel_refl, scaling factor for reflection; if < 0, no direct component (rel_refl=1 for isotropic source above disk)
`par4`	=	redshift
`par5`	=	abundance of elements heavier than He relative to that defined by abund
`par6`	=	iron abundance relative to that defined by abund
`par7`	=	cosine of inclination angle
`K`	=	photon flux at 1 keV (photons/keV/cm^2/s) of the power-law only in the observed frame.

6.2.43 pexriv

Exponentially cut off power law spectrum reflected from ionized material (Magdziarz & Zdziarski MNRAS, 273, 837; 1995). Ionization and opacities of the reflecting medium is computed as in the procedure absori. The output spectrum is the sum of the cutoff power law and the reflection component. The reflection component alone can be obtained for rel_refl < 0. Then the actual reflection normalization is $|$rel_refl$|$. Note that you need to change then the limits of rel_refl excluding zero (as then the direct component appears). If $E_c = 0$ there is no cutoff in the power law. The metal and iron abundances are variable with respect to those defined by the command abund. Send questions or comments to aaz@camk.edu.pl.

`par1`	=	γ, power law photon index, N_E prop. to $E^{-\gamma}$
`par2`	=	E_c, the cutoff energy in keV (if $E_c = 0$ there is no cutoff; one needs to change the lower limit for that)
`par3`	=	rel_refl, scaling factor for reflection; if < 0, no direct component (rel_refl=1 for isotropic source above disk)
`par4`	=	redshift, z
`par5`	=	abundance of elements heavier than He relative to that defined by abund
`par6`	=	iron abundance relative to that defined by abund
`par7`	=	cosine of inclination angle
`par8`	=	disk temperature in K

par9 = disk ionization parameter, $xi = 4\pi F_{ion}/n$, where F_{ion} is the 5eV – 20keV irradiating flux, n is the density of the reflector; see Done et al., 1992, ApJ, 395, 275

K = photon flux at 1 keV (photons/keV/cm^2/s) of the power-law only in the observed frame.

6.2.44 plcabs

This model describes X-ray transmission of an isotropic source of photons located at the center of a uniform, spherical distribution of matter, correctly taking into account Compton scattering. The model can be used for radial column densities up to 5×10^{24} cm^{-2}. The valid energy range for which data can be mod!laeled is between 10 and 18.5 keV, depending on the column density. Details of the physics of the model, the approximations used and further details on the regimes of validity can be found in Yaqoob (1997; ApJ, 479, 184). In this particular incarnation, the initial spectrum is a power law modified by a high-energy exponential cut-off above a certain threshold energy.

Also, to improve the speed, a FAST option is available in which a full integration over the input spectrum is replaced by a simple mean energy shift for each bin. This option is obtained by setting parameter 9 to a value of 1 or greater and MUST BE FIXED. Further, for single-scattering albedos less than ACRIT (i.e. parameter 8) energy shifts are neglected altogether. The recommended value is ACRIT=0.1 which corresponds to about 4 keV for cosmic abundances and is more than adequate for ASCA data.

Note that for column densities in the range $10^{23} - 10^{24}$ cm^{-2}, the maximum number of scatterings which need be considered for convergence of the spectrum of better than 1% is between 1 and 5. For columns as high as 5×10^{24}, the maximum number of scatterings which need be considered for the same level of convergence is 12. **NOTE THAT NMAX MUST BE FROZEN **

par1 = Column density in units 10^{22} cm^{-2}.
par2 = Maximum number of scatterings to consider.
par3 = Iron abundance.
par4 = Iron K edge energy.
par5 = Power-law photon index.
par6 = High-energy cut-off threshold energy.
par7 = High-energy cut-off e-folding energy.
par8 = Critical albedo for switching to elastic scattering.
par9 = If `par9` > 1 function uses mean energy shift, not integration.
par10 = Source redshift.

6.2.45 posm

Positronium continuum (Brown & Leventhal 1987 ApJ 319, 637).

$$A(E) = K(2.0/((3.14159^2 - 9.0)511))(E(511-E)/(1022-E)^2 + (1022(511-E)/E^2)\log((511-E)/511) - (1022(511-E)^2/(1022-E)^3)\log((511-E)/511) + (1022-E)/E)$$

for $E < 511 keV$, where :

K = normalization.

6.2.46 powerlaw

Simple photon power law.

$$A(E) = \mathtt{K}(E/1\text{keV})^{-\mathtt{par1}}$$

where :

par1	=	photon index of power law (dimensionless)
K	=	photons/keV/cm^2/s at 1 keV.

6.2.47 pshock

Constant temperature, plane-parallel shock plasma model. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

par1	=	plasma temperature (keV)
par2	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni. Abundances are given by the Anders & Grevesse mixture.
par3	=	Lower limit on ionization timescale (s/cm^3) to include.
par4	=	Upper limit on ionization timescale (s/cm^3) to include.
par5	=	redshift z
K	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.48 raymond

An emission spectrum from hot, diffuse gas based on the model calculations of Raymond and Smith (ApJSuppl 35, 419 and additions) including line emissions from several elements. This model interpolates on a grid of spectra for different temperatures. The grid is logarithmically spaced with 80 temperatures ranging from 0.008 to 80 keV.

par1	=	plasma temperature in keV
par2	=	Metal abundances (He fixed at cosmic) The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni and their relative abundances are set by the `abund` command.
par3	=	redshift
K	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_h dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.49 redge

Recombination edge emission.

$$\begin{aligned} A(E) &= 0. && \text{for } E < \mathtt{par1} \\ A(E) &= K(1/\mathtt{par2})\exp(-(E-\mathtt{par1})/\mathtt{par2}) && \text{for } E > \mathtt{par1} \end{aligned}$$

`par1`	=	threshold energy
`par2`	=	plasma temperature (keV)
`K`	=	total photons/cm^2/s in the line

6.2.50 refsch

Exponentially cut-off power-law spectrum reflected from an ionized relativistic accretion disk. In this model, spectrum of `pexriv` is convolved with a relativistic disk line profile `diskline`. See Magdziarz & Zdziarski 1995 MNRAS, 273, 837 for details of Compton reflection. See Fabian et al. 1989, MNRAS, 238, 729 for details of the disk line profile.

`par1`	=	Γ, power law photon index, N_E prop. to $E^{-\Gamma}$
`par2`	=	E_c, the cutoff energy in keV (if $E_c = 0$ there is no cutoff)
`par3`	=	rel_{refl}, reflection scaling factor (1 for isotropic source above disk)
`par4`	=	redshift, z
`par5`	=	abundance of elements heavier than He relative to the solar abundances
`par6`	=	iron abundance relative to the above
`par7`	=	inclination angle (degrees)
`par8`	=	disk temperature in K
`par9`	=	disk ionization parameter, $\xi = 4\pi F_{ion}/n$, where F_{ion} is the 5 eV - 20 keV irradiating flux, n is the density of the reflector; see Done et al., 1992, ApJ, 395, 275
`par10`	=	power law index for reflection emissivity; emissivity is $\propto r^{\texttt{par10}}$
`par11`	=	inner disk radius in units of GM/c^2
`par12`	=	outer disk radius in units of GM/c^2
`par13`	=	internal model accurancy - points of spectrum per energy decade
`K`	=	photon flux at 1 keV of the cutoff broken power-law only (no reflection) in the observed frame.

6.2.51 sedov

Sedov model with separate ion and electron temperatures. This model is slow. `par1` provides a measure of the average energy per particle (ions+electrons) and is constant throughout the postshock flow in plane shock models (Borkowski et al., 2001, ApJ, 548, 820). `par2` should always be less than `par1`. If `par2` exceeds `par1` then their interpretations are switched (ie the larger of `par1` and `par2` is always the mean temperature). Additional references can be found under the help for the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	mean shock temperature (keV)
`par2`	=	electron temperature immediately behind the shock front (keV).
`par3`	=	Metal abundances (He fixed at cosmic). The elements included are C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni. Abundances are given by the Anders & Grevesse mixture.

`par4`	=	ionization age (s/cm^3) of the remnant (== electron density immediately behind the shock front times age of remnant)
`par5`	=	redshift z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2))\int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.52 srcut

The synchrotron spectrum from an exponentially cut off power-law distribution of electrons in a homogeneous magnetic field. This spectrum is itself a power-law, rolling off more slowly than exponential in photon energies. Though more realistic than a power-law, it is highly oversimplified, but does give the maximally curved physically plausible spectrum and can be used to set limits on maximum accelerated-electron energies even in remnants whose X-rays are thermal. See Reynolds, S.P. & Keohane, J.W. 1999, ApJ, 525, 368 and Reynolds, S.P., 1998 ApJ 493, 357. Note that the radio spectral index and flux can be obtained from Green's Catalogue at `http://www.mrao.cam.ac.uk/surveys/snrs/` for galactic SNRs.

`par1`	=	alpha: radio spectral index
`par2`	=	break Hz: approximately the frequency at which the flux has dropped by a factor of 10 from a straight powerlaw.
`K`	=	1 GHz flux (Jy)

6.2.53 sresc

The synchrotron spectrum from an electron distribution limited by particle escape above some energy. The electrons are shock-accelerated in a Sedov blast wave encountering a constant-density medium containing a uniform magnetic field. The model includes variations in electron acceleration efficiency with shock obliquity, and post-shock radiative and adiabatic losses, as described in Reynolds, S.P., ApJ 493, 357 1998. This is a highly specific, detailed model for a fairly narrow set of conditions. See also Reynolds, S.P., ApJL 459, L13 1996. Note that the radio spectral index and flux can be obtained from Green's Catalogue at `http://www.mrao.cam.ac.uk/surveys/snrs/` for galactic SNRs.

`par1`	=	alpha: radio spectral index (flux proportional to frequency $f^{-\alpha}$)
`par2`	=	break Hz: approximately the frequency at which the flux has dropped by a factor of 6 below a straight powerlaw extrapolation from radio frequencies. This frequency is 5.3 times the peak frequency radiated by electrons with energy E_{m3} in a magnetic field of 4 B_1, in the notation of Reynolds (1998), Eq. (19).
`K`	=	1 GHz flux (Jy)

6.2.54 step

A step function convolved with a gaussian.

$$N(E) = \mathtt{K}(1 - \mathrm{erf}((E - \mathtt{par1})/\sqrt{(2)}/\mathtt{par2}))/2$$

`par1`	=	start energy (keV)

`par2`	=	gaussian sigma (keV)
`K`	=	step amplitude

6.2.55 vapec

An emission spectrum from collisionally-ionized diffuse gas calculated using the APEC code v1.0. More information can be found at `http://hea-www.harvard.edu/APEC/` which should be consulted by anyone running this model. If the environment variable `XSPEC_APEC` is set then the model uses the filename `XSPEC_APEC` concatenated with _coco.fits and _line.fits instead of the standard APEC data files.

`par1`	=	plasma temperature in keV
`par2—par14`	=	Abundances for He, C, N, O, Ne, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par15`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.56 vbremss

A thermal bremsstrahlung spectrum with variable He/H. Based on the Kellogg, Baldwin & Koch polynomial (ApJ 199, 299) fits to the Karzas & Latter (ApJSuppl 6, 167) numerical values. A routine from Kurucz (priv.comm.) is used for low temperatures.

`par1`	=	plasma temperature (keV)
`par2`	=	n(He)/n(H) (note that the Solar ratio is 0.085)
`K`	=	$(3.02 \times 10^{-15}/(4\pi D^2)) \int n_e n_I dV$, where D is the distance to the source (cm) and n_e, n_I are the electron and ion densities (cm^{-3})

6.2.57 vequil

Ionization equilibrium collisional plasma model. This is the equilibrium version of Kazik Borkowski's NEI models. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature in keV
`par2—par13`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (given by the Anders & Grevesse mixture)
`par14`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.58 vgnei

Non-equilibrium ionization collisional plasma model. This is a generalization of the nei model where the temperature is allowed to have been different in the past ie the ionization timescale averaged temperature is not necessarily equal to the current temperature. For example, in a standard Sedov model with equal electron and ion temperatures, the ionization timescale averaged temperature is always higher than the current temperature for each fluid element. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature in keV
`par2`	=	H density in cm^{-3}
`par3—par14`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (given by the Anders & Grevesse mixture)
`par15`	=	Ionization timescale in units of s/cm^{-3}
`par16`	=	Ionization timescale averaged plasma temperature in keV.
`par17`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.59 vmeka

An emission spectrum from hot diffuse gas based on the model calculations of Mewe and Gronenschild (as amended by Kaastra - for references see the section on the meka model). The model includes line emissions from several elements. Abundances are the number of nuclei per Hydrogen nucleus relative to the Solar abundances set by the `abund` command.

`par1`	=	plasma temperature in keV
`par2`	=	hydrogen density in cm^{-3}
`par3—par16`	=	Abundances for He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par17`	=	redshift
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_h dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.60 vmekal

An emission spectrum from hot diffuse gas based on the model calculations of Mewe and Kaastra with Fe L calculations by Liedahl (for references see the section on the mekal model). The model includes line emissions from several elements. Abundances are the number of nuclei per Hydrogen nucleus relative to the Solar abundances set by the abund command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate.

`par1`	=	plasma temperature in keV
`par2`	=	hydrogen density in cm^{-3}
`par3—par16`	=	Abundances for He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par17`	=	redshift
`par18`	=	0 $\Rightarrow$ calculate, 1 $\Rightarrow$ interpolate
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_h dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.61 vmcflow

A cooling flow model after Mushotzky & Szymkowiak ("Cooling Flows in Clusters and Galaxies" ed. Fabian, 1988). This one uses the `vmekal` model for the individual temperature components, but is otherwise identical to `mkcflow`. Abundances are relative to Solar as set by the `abund` command. The switch parameter determines whether the mekal code will be run to calculate the model spectrum for each temperature or whether the model spectrum will be interpolated from a pre-calculated table. The former is slower but more accurate.

`par1`	=	low temperature (keV)
`par2`	=	high temperature (keV)
`par3-par16`	=	Abundances for He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par17`	=	redshift
`par18`	=	0 $\Rightarrow$ calculate, 1 $\Rightarrow$ interpolate
`K`	=	Mass accretion rate (solar mass/yr)

6.2.62 vnei

Non-equilibrium ionization collisional plasma model. This assumes a constant temperature and single ionization parameter. It provides a characterisation of the spectrum but is not a physical model. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature in keV
`par2`	=	H density in cm^{-3}
`par3—par14`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (given by the Anders & Grevesse mixture)
`par15`	=	Ionization timescale in units of s/cm^{-3}
`par16`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.63 vpnshock

Plane-parallel shock plasma model with separate ion and electron temperatures. This model is slow. `par1` provides a measure of the average energy per particle (ions+electrons) and is constant throughout the post-shock flow in plane shock models (Borkowski et al., 2001, ApJ, 548, 820). `par2` should always be less than `par1`. If `par2` exceeds `par1` then their interpretations are switched (ie the larger of `par1` and `par2` is always the mean temperature). Additional references can be found under the help for the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	mean shock temperature in keV
`par2`	=	electron temperature immediately behind the shock front (keV)
`par3`	=	H density in cm^{-3}
`par4`—`par15`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (given by the Anders & Grevesse mixture)
`par16`	=	Lower limit on ionization timescales (s/cm^3) to include
`par17`	=	Upper limit on ionization timescales (s/cm^3) to include
`par18`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2))\int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.64 vpshock

Constant temperature, plane-parallel shock plasma model. The references for this model can be found under the description of the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	plasma temperature in keV
`par2`	=	H density in cm^{-3}
`par3`—`par14`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (given by the Anders & Grevesse mixture)
`par15`	=	Lower limit on ionization timescales (s/cm^3) to include
`par16`	=	Upper limit on ionization timescales (s/cm^3) to include
`par17`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2))\int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.65 vraymond

A version of the XSPEC model `raymond`, with independent variables for all abundances and for the continuum. This model interpolates on a grid of spectra for different temperatures. The grid is logarithmically spaced with 80 temperatures ranging from 0.008 to 80 keV. Abundances are the number of nuclei per Hydrogen nucleus relative to the Solar abundances as set by the `abund` command.

`par1`	=	plasma temperature (keV)
`par2—par13`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (defined by the abund command)
`par14`	=	redshift
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_h dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.66 vsedov

Sedov model with separate ion and electron temperatures. This model is slow. `par1` provides a measure of the average energy per particle (ions+electrons) and is constant throughout the postshock flow in plane shock models (Borkowski et al., 2001, ApJ, 548, 820). `par2` should always be less than `par1`. If `par2` exceeds `par1` then their interpretations are switched (ie the larger of `par1` and `par2` is always the mean temperature). Additional references can be found under the help for the equil model. Several versions are available. To switch between them use the `xset neivers` command. `xset neivers 1.0` gives the version from xspec v11.1, `xset neivers 1.1` uses updated calculations of ionization fractions using dielectronic recombination rates from Mazzotta et al (1988), and `xset neivers 2.0` uses the same ionization fractions as 1.1 but uses APED to calculate the resulting spectrum. Note that versions 1.x have no emission from Ar. The default is version 1.1.

`par1`	=	mean shock temperature in keV
`par2`	=	electron temperature immediately behind the shock front (keV)
`par3`	=	H density in cm^{-3}
`par4—par15`	=	Abundances for He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, Ni wrt Solar (given by the Anders & Grevesse mixture)
`par16`	=	ionization age (s/cm^3) of the remnant (== electron density immediately behind the shock front times age of remnant)
`par17`	=	redshift, z
`K`	=	$(10^{-14}/(4\pi(D_A(1+z))^2)) \int n_e n_H dV$, where D_A is the angular size distance to the source (cm), n_e and n_H are the electron and H densities (cm^{-3})

6.2.67 zbbody

A redshifted blackbody spectrum.

$$A(E) = \mathtt{K}\ 8.0525(E(1+\mathtt{par2}))^2/((kT)^4(\exp(E(1+\mathtt{par2})/\mathtt{par1})-1))$$

where :

`par1`	=	temperature kT in keV
`par2`	=	redshift
`K`	=	L_{39}/D_{10}^2, where L39 is the source luminosity in units of 10^{39} ergs/sec and D_{10} is the angular size distance to the source in units of 10 kpc

6.2.68 zbremss

A redshifted thermal bremsstrahlung spectrum based on the Kellogg, Baldwin & Koch (ApJ 199, 299) polynomial fits to the Karzas & Latter (ApJSuppl 6, 167) numerical values. A routine from Kurucz (priv.comm.) is used for low temperatures.

`par1`	=	plasma temperature in keV
`par2`	=	redshift
`K`	=	$(3.02 \times 10^{-15}/(4\pi D^2)) \int n_e n_I dV$, where D is the distance to the source (cm) and n_e, n_I are the electron and ion densities (cm^{-3})

6.2.69 zedge

A redshifted absorption edge.

$$\begin{aligned} M(E) &= 1 && \text{for } E < \texttt{par1} \\ &= \exp(-\texttt{par2}((E(1+\texttt{par3})/\texttt{par1})^3)) && \text{for } E > \texttt{par1} \end{aligned}$$

where :

`par1`	=	threshold energy
`par2`	=	absorption depth at threshold
`par3`	=	redshift

6.2.70 zgauss

A redshifted simple gaussian line profile. If the width is ≤ 0, then it is treated as a delta function.

$$A(E) = \texttt{K}(1./(\sqrt{(}2\pi\texttt{par2}^2)) \exp(-.5((E(1+\texttt{par3}) - \texttt{par1})/\texttt{par2})^2))$$

where :

`par1`	=	line energy in keV
`par2`	=	line width (sigma) in keV
`par3`	=	redshift
`K`	=	total photons/cm^2/s in the line

6.2.71 zpowerlw

A simple photon power law.

$$A(E) = \texttt{K}(E(1+\texttt{par2})/(1\text{keV}))^{(-\texttt{par1})}$$

where :

`par1`	=	photon index of power law (dimensionless)
`par2`	=	redshift
`K`	=	photons/keV/cm^2/s at 1 keV

6.3 Multiplicative Model Components

Multiplicative, convolution, and mixing models represent media intervening between sources and the observer that modify the source flux in an energy-dependent way.

6.3.1 absori

An ionized absorber based on that of Done et al. (1992, ApJ 395, 275) and developed by Magdziarz & Zdziarski. See also Zdziarski et al. (1995, ApJ 438, L63). Photoionization rates are from Reilman & Manson (1979, ApJS 40, 815), who employ the Hartree-Slater approximation (accurate to about 5%), and recombination rates are from Shull & Steenburg (1982, ApJS 48, 95). The cross sections are extrapolated with E^{-3} above 5 keV. The abundances are set up by the command abund. Send questions or comments to aaz@camk.edu.pl.

`par1`	=	power-law photon index.
`par2`	=	Hydrogen column in units of 10^{22} cm^{-2}.
`par3`	=	Absorber temperature in K.
`par4`	=	Absorber ionization state (L/nR^2), see Done et al. (1992).
`par5`	=	Redshift.
`par6`	=	Iron abundance relative to that defined by the command abund.

6.3.2 cabs

Non-relativistic, optically-thin Compton scattering.

$$M(E) = \exp(-\mathtt{par1}\sigma(E))$$

where $\sigma(E)$ is the Thomson cross-section.

`par1`	=	hydrogen column (in units of 10^{22} atoms/cm^2)

6.3.3 constant

An energy-independent multiplicative factor.

`par1`	=	factor

6.3.4 cyclabs

A cyclotron absorption line as used in pulsar spectra. See Mihara et al., *Nature*, 1990 or Makishima et al. *PASJ*, 1990.

$$\begin{aligned} M(E) &= \exp(-\mathtt{par1}(\mathtt{par3}E/\mathtt{par2})^2/((E-\mathtt{par2})^2+\mathtt{par3}^2) \\ &\quad +\mathtt{par4}(\mathtt{par5}E/(2\mathtt{par2}))^2/((E-2\mathtt{par2})^2+\mathtt{par5}^2))) \end{aligned}$$

`par1`	=	depth of the fundamental
`par2`	=	cyclotron energy
`par3`	=	width of the fundamental
`par4`	=	depth 2nd harmonic

par5	=	width of the 2nd harmonic

6.3.5 dust

A modification of a spectrum due to scattering off dust on the line-of-sight. The model assumes that the scattered flux goes into a uniform disk whose size has a 1/E dependence and whose total flux has a 1/E^2 dependence.

par1	=	scattering fraction at 1 keV
par2	=	size of halo at 1 keV in units of the detector beamsize

6.3.6 edge

An absorption edge.

$$\begin{aligned} M(E) &= 1 & \text{for } E \leq \texttt{par1} \\ &= \exp(-\texttt{par2}((E/\texttt{par1})^{-3})) & \text{for } E \geq \texttt{par1} \end{aligned}$$

where :

par1	=	threshold energy
par2	=	absorption depth at the threshold

6.3.7 etable

An exponential table model. The filename to be used should be given immediately after etable in the model command. For example

```
XSPEC> model etable{mymod.mod}
```

uses mymod.mod as the input for the model. XSPEC will multiply the contents of the model by –1 then take the exponential ie this model is for calculating absorption functions. For specifications of the table model file, see the OGIP memo 92-009 on the FITS file format for table model files (available on the WWW or by anonymous ftp from `ftp://legacy.gsfc.nasa.gov/caldb/docs/memos`).

6.3.8 expabs

A low-energy exponential rolloff.

$$M(E) = \exp(-\texttt{par1}/E)$$

where :

par1	=	e-folding energy for the absorption

6.3.9 expfac

An exponential modification of a spectrum.

$$\begin{aligned} M(E) &= 1. + \texttt{par1}\exp(-\texttt{par2}(E/1\mathrm{keV})) & \text{for } E > \texttt{par3} \\ M(E) &= 1. & \text{for } E < \texttt{par3} \end{aligned}$$

`par1`	=	amplitude of effect
`par2`	=	exponential factor
`par3`	=	start energy of modification

6.3.10 highecut

A high energy cutoff.

$$
\begin{aligned}
A(E) &= \exp((\texttt{par1} - E)/\texttt{par2}) && \text{for } E \geq \texttt{par1} \\
A(E) &= 1 && \text{for } E \leq \texttt{par1}
\end{aligned}
$$

where :

`par1`	=	cutoff energy in keV
`par2`	=	e-folding energy in keV

6.3.11 hrefl

A simple multiplicative reflection model due to Tahir Yaqoob. Parameters are as follows:

`par1`	=	minimum angle (degrees) between source photons incident on the slab and the slab normal (=arctan(Ri/H).
`par2`	=	maximum angle (degrees) between source photons incident on the slab and the slab normal (=arctan(Ro/H).
`par3`	=	Angle (degrees) between the observer's line of sight and the slab normal.
`par4`	=	Iron abundance relative to Solar.
`par5`	=	Iron K-edge energy.
`par6`	=	Fraction of the direct flux seen by the observer.
`par7`	=	Normalization of the reflected continuum.
`par8`	=	redshift.

This model gives the reflected X-ray spectrum from a cold, optically thick, circular slab with inner and outer radii (Ri & Ro, respectively) illuminated by a point source a height H above the centre of the slab. The main difference between this model and other reflection models is that analytic approximations are used for the Chandrasekar H functions (and their integrals) and ELASTIC SCATTERING is assumed (see Basko 1978, *ApJ*, 223, 268). The elastic-scattering approximation means that the model is ONLY VALID UP TO $\approx$15 keV in the source frame. Future enhancements will include fudge factors that will allow extension up to 100 keV. The fact that no integration is involved at any point makes the routine very fast and particularly suitable for generating error contours, especially when fitting a large number of data channels. The model is multiplicative, and so can be used with ANY incident continuum.

Suppose the incident photon spectrum is N(E) photons/cm/cm/s/keV and that the incident continuum is steady in time, and suppose further that the reflected continuum from the slab is R(E). When you multiply the incident spectrum with HREFL, what you actually get is the following:

$$
model(E) = \texttt{par6}N(E) + \texttt{par7}R(E)
$$

Thus, the actual physical situation described above corresponds to `par6`=1.0 and `par7`=1.0. You may decide to float `par6` and/or `par7`. In that case, you must decide what the best-fitting values of these parameters mean physically for your case. It may imply time-lags between the direct and reflected components, different source and/or disk geometries to those assumed, or something else.

6.3.12 mtable

A multiplicative table model. The filename to be used should be given immediately after `mtable` in the `model` command. For example

```
XSPEC> model mtable{mymod.mod} ...
```

uses `mymod.mod` as the input for the model. For specifications of the table model file, see the OGIP memo 92-009 on the FITS file format for table model files (available on the WWW or by anonymous ftp from `ftp://legacy.gsfc.nasa.gov/caldb/docs/memos`). An example multiplicative table model file is testpcfabs.mod in $XANADU/src/spectral/session.

6.3.13 notch

A notch line absorption. This is model is equivalent to a very saturated absorption line.

$$\begin{aligned} M(E) &= (1 - \texttt{par3}) \quad \text{for } \texttt{par1} - \texttt{par2}/2 < E < \texttt{par1} + \texttt{par2}/2 \\ &= 1 \quad \text{for all other} \end{aligned}$$

where :

`par1`	=	line energy (keV)
`par2`	=	line width (keV)
`par3`	=	covering fraction

6.3.14 pcfabs

A partial covering fraction absorption. The relative abundances are set by the `abund` command.

$$M(E) = \texttt{par2}\exp(-\texttt{par1}\sigma(E)) + (1 - \texttt{par2})$$

where σ(E) is the photo-electric cross-section (NOT including Thomson scattering) (see `phabs`) and:

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	covering fraction (0 < par2 <= 1.) (dimensionless)

6.3.15 phabs

A photoelectric absorption using cross-sections set by the xsect command. The relative abundances are set by the abund command.

$$M(E) = \exp(-\texttt{par1}\sigma(E))$$

where $\sigma(E)$ is the photo-electric cross-section (NOT including Thomson scattering). Note that the default He cross-section changed in v11. The old version can be recovered using the command xsect obcm.

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)

6.3.16 plabs

Absorption as a power-law in energy. Useful for things like dust.

$$M(E) = \texttt{par2}(E/1\mathrm{keV})^{-\texttt{par1}}$$

`par1`	=	index
`par2`	=	coefficient

6.3.17 redden

IR/optical/UV extinction from Cardelli et al. (1989, ApJ, 345, 245). The transmission is set to unity shortward of the Lyman limit. This is incorrect physically but does allow the model to be used in combination with an X-ray photoelectric absorption model such as phabs.

`par1`	=	E(B-V)

6.3.18 smedge

A smeared edge (Ebisawa PhD thesis, implemented by Frank Marshall).

$$\begin{aligned} M(E) &= 1. && \text{for } E \leq \texttt{par1} \\ M(E) &= \exp(-\texttt{par2}((E/\texttt{par1})^{\texttt{par3}})(1-\exp((\texttt{par1}-E))/\texttt{par4})) && \text{for } E \geq \texttt{par1} \end{aligned}$$

`par1`	=	the threshold energy (keV)
`par2`	=	the maximum absorption factor at threshold
`par3`	=	index for photo-electric cross-section (normally -2.67)
`par4`	=	smearing width (keV)

6.3.19 spline

A cubic spline modification.

`par1`	=	start x-value
`par2`	=	start y-value
`par3`	=	end y-value
`par4`	=	start dy/dx
`par5`	=	end dy/dx
`par6`	=	end x-value

6.3.20 SSS ice

The Einstein Observatory SSS ice absorption.

`par1`	=	ice thickness parameter

6.3.21 tbabs

The Tuebingen-Boulder ISM absorption model. This model calculates the cross section for X-ray absorption by the ISM as the sum of the cross sections for X-ray absorption due to the gas-phase ISM, the grain-phase ISM, and the molecules in the ISM. In the grain-phase ISM, the effect of shielding by the grains is accounted for, but is extremely small. In the molecular contribution to the ISM cross section, only molecular hydrogen is considered. In the gas-phase ISM, the cross section is the sum of the photoionization cross sections of the different elements, weighted by abundance and taking into account depletion onto grains. In addition to the updates to the photoionization cross sections, the gas-phase cross section differs from previous values as a result of updates to the ISM abundances. These updated abundances are available through the `abund wilm` command. Details of updates to the photoionization cross sections as well as to abundances can be found in Wilms, Allen and McCray (2000, ApJ 542, 914).

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)

6.3.22 tbgrain

The Tuebingen-Boulder ISM absorption model. This model calculates the cross section for X-ray absorption by the ISM as the sum of the cross sections for X-ray absorption due to the gas-phase ISM, the grain-phase ISM, and the molecules in the ISM. In the grain-phase ISM, the effect of shielding by the grains is accounted for, but is extremely small. In the molecular contribution to the ISM cross section, only molecular hydrogen is considered. In the gas-phase ISM, the cross section is the sum of the photoionization cross sections of the different elements, weighted by abundance and taking into account depletion onto grains. In addition to the updates to the photoionization cross sections, the gas-phase cross section differs from previous values as a result of updates to the ISM abundances. These updated abundances are available through the `abund wilm` command. Details of updates to the photoionization cross sections as well as to abundances can be found in Wilms, Allen and McCray (2000, ApJ 542, 914). This model allows the user to vary the molecular hydrogen column and the grain distribution parameters.

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	molecular hydrogen column (in units of 10^{22} atoms/cm^2)
`par3`	=	grain density (in gm/cm^3)
`par4`	=	grain minimum size (in micron)
`par5`	=	grain maximum size (in micron)
`par6`	=	power-law index of grain sizes

6.3.23 tbvarabs

The Tuebingen-Boulder ISM absorption model. This model calculates the cross section for X-ray absorption by the ISM as the sum of the cross sections for X-ray absorption due to the gas-phase ISM, the grain-phase ISM, and the molecules in the ISM. In the grain-phase ISM, the effect of shielding by the grains is accounted for, but is extremely small. In the molecular contribution to the ISM cross section, only molecular hydrogen is considered. In the gas-phase ISM, the cross section is the sum of the photoionization cross sections of the different elements, weighted by abundance and taking into account depletion onto grains. In addition to the updates to the photoionization cross sections, the gas-phase cross section differs from previous values as a result of updates to the ISM abundances. These updated abundances are available through the `abund wilm` command. Details of updates to the photoionization cross sections as well as to abundances can be found in Wilms, Allen and McCray (2000, ApJ 542, 914). This model allows the user to vary the molecular hydrogen column, the grain distribution parameters, and the abundances and grain depletions.

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2` – `par18`	=	abundance (relative to Solar) of He, C, N, O, Ne, Na, Mg, Al, Si, S, Cl, Ar, Ca, Cr, Fe, Co, Ni
`par19`	=	molecular hydrogen column (in units of 10^{22} atoms/cm^2)
`par20`	=	grain density (in gm/cm^3)
`par21`	=	grain minimum size (in micron)
`par22`	=	grain maximum size (in micron)
`par23`	=	power-law index of grain sizes
`par24` – `par41`	=	grain depletion fractions of He, C, N, O, Ne, Na, Mg, Al, Si, S, Cl, Ar, Ca, Cr, Fe, Co, Ni
`par42`	=	redshift

6.3.24 uvred

A UV reddening using Seaton's law (*M.N.R.A.S.*, 187, 75p). Valid from 1000-3704Å. The transmission is set to unity shortward of the Lyman limit. This is incorrect physically but does allow the model to be used in combination with an X-ray photoelectric absorption model such as phabs.

`par1`	=	E(B-V)

6.3.25 varabs

A photoelectric absorption with variable abundances using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. The column for each element is in units of the column in a solar abundance column of an equivalent hydrogen column density of 10^{22} cm^{-2}. The Solar abundance table used is set by the abund command.

`par1`–`par18`	=	equivalent columns for H, He, C, N, O, Ne, Na, Mg, Al, Si, S, Cl, Ar, Ca, Cr, Fe, Co, Ni

6.3.26 vphabs

A photoelectric absorption with variable abundances using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. The relative abundances are set by the abund command. This model is identical to `varabs` except for the way that the parameters are defined.

$$M(E) = \exp(-\mathtt{par1}\sigma(E))$$

where $\sigma(E)$ is the photo-electric cross-section (NOT including Thomson scattering) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2-par18`	=	abundances for He, C, N, O, Ne, Na, Mg, Al, Si, S, Cl, Ar, Ca, Cr, Fe, Co, Ni wrt to Solar (defined by the abund command)

6.3.27 wabs

A photo-electric absorption using Wisconsin (Morrison and McCammon; ApJ 270, 119) cross-sections.

$$M(E) = \exp(-\mathtt{par1}\sigma(E))$$

where σ(E) is the photo-electric cross-section (NOT including Thomson scattering). Note that this model uses the Anders & Ebihara relative abundances (1982, Geochimica et Cosmochimica Acta 46, 2363) regardless of the `abund` command.

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)

6.3.28 wndabs

Photo-electric absorption from approximation to a warm absorber using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. Relative abundances are set by the `abund` command.

$$\begin{aligned} M(E) &= \exp(-\mathtt{par1}\sigma(E)) && \text{for } E \geq \mathtt{par2} \\ &= 1. && \text{for } E \leq \mathtt{par2} \end{aligned}$$

where σ(E) is the photo-electric cross-section (NOT including Thomson scattering) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	window energy (keV)

6.3.29 xion

This model describes the reflected spectra of a photo-ionized accretion disk or a ring if one so chooses. The approach is similar to the one used for tables with stellar spectra. Namely, a large number of models are computed for a range of values of the spectral index, the incident X-ray flux, disk gravity, the thermal disk flux and iron abundance. Each model's output is an un-smeared reflected spectrum for 5 different inclination angles ranging from nearly pole-on to nearly face on, stored in a look-up table. The default geometry is that of a lamppost, with free parameters of the model being the height of the X-ray source above the disk, h_X, the dimensionless accretion rate through the disk, $\dot{m}$, the luminosity of the X-ray source, L_X, the inner and outer disk radii, and the spectral index. This defines the gravity parameter, the ratio of X-ray to thermal fluxes, etc., for each radius, which allows the use of a look-up table to approximate the reflected spectrum. This procedure is repeated for about 30 different radii. The total disk spectrum is then obtained by integrating over the disk surface, including relativistic smearing of the spectrum for a non-rotating black hole (e.g., Fabian 1989).

In addition, the geometry of a central sphere (with power-law optically thin emissivity inside it) plus an outer cold disk, and the geometry of magnetic flares are available (param(13)=2 and 3, respectively). One can also turn off relativistic smearing to see what the local disk spectrum looks like (param(12) = 2 in this case; otherwise leave it at 4). In addition, param(11)=1 produces reflected plus direct spectrum/direct; param(11)=2 produces (incident + reflected)/incident [note that normalization of incident and direct are different because of solid angles covered by the disk; 2 should be used for magnetic flare model]; and param(11)=3 produces reflected/incident. Abundance is controlled by param(9) and varies between 1 and 4 at the present. A much more complete description of the model will be presented in Nayakshin et al. 2001 (currently a draft is available at http://lheawww.gsfc.nasa.gov/users/serg/ms.ps)

`par1`	=	height of the source above the disk (in Schwarzschild radii)
`par2`	=	ratio of the X-ray source luminosity to that of the disk
`par3`	=	accretion rate (in Eddington units)
`par4`	=	$\cos i$, the inclination angle (1 = face-on)
`par5`	=	inner radius of the disk (in Schwarzschild radii)
`par6`	=	outer radius of the disk (in Schwarzschild radii)
`par7`	=	photon index of the source
`par8`	=	redshift z
`par9`	=	Fe abundance relative to Solar (which is defined as 3.16×10^{-5} by number relative to H)
`par10`	=	Exponential high energy cut-off energy for the source
`par11`	=	1 $\Rightarrow$ (reflected+direct)/direct, 2 $\Rightarrow$ (reflected+incident)/incident, 3 $\Rightarrow$ reflected/incident
`par12`	=	2 $\Rightarrow$ no relativistic smearing, 4 $\Rightarrow$ relativistic smearing
`par13`	=	1 $\Rightarrow$ lamppost, 2 $\Rightarrow$ central hot sphere with outer cold disk, 3 $\Rightarrow$ magnetic flares above a cold disk. Note that setting par13 to 2.y gives a central hot sphere with luminosity law $dL/dR = 4\pi R^2 R^{-10y}$. The inner radius of the sphere is 3 Schwarzschild radii and the outer radius is equal to par1. Only the case with p$ar5$ $>=$ p$ar1$ has been tested so far.

6.3.30 zhighect

A redshifted high energy cutoff.

$$
\begin{aligned}
A(E) &= \exp((\mathtt{par1} - E(1+\mathtt{par3}))/\mathtt{par2}) && \text{for } E > \mathtt{par1} \\
A(E) &= 1 && \text{for } E < \mathtt{par1}
\end{aligned}
$$

where :

`par1`	=	cutoff energy in keV
`par2`	=	e-folding energy in keV
`par3`	=	redshift

6.3.31 zpcfabs

A redshifted partial covering fraction absorption. Relative abundances are set by the `abund` command.

$$M(E) = \texttt{par2} \exp(-\texttt{par1}\sigma(E(1+\texttt{par3}))) + (1-\texttt{par2})$$

where σ(E) is the photo-electric cross-section (NOT including Thomson scattering) (see `phabs`) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	covering fraction ($0 <$ `par2` $\leq 1.$) (dimensionless)
`par3`	=	redshift

6.3.32 zphabs

A redshifted photoelectric absorption using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. The relative abundances are set by the abund command.

$$M(E) = \exp(-\texttt{par1}\sigma(E(1+\texttt{par2}))$$

where $\sigma(E)$ is the photo-electric cross-section (NOT including Thomson scattering) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	redshift

6.3.33 ztbabs

The Tuebingen-Boulder ISM absorption model. This model calculates the cross section for X-ray absorption by the ISM as the sum of the cross sections for X-ray absorption due to the gas-phase ISM and the molecules in the ISM. In the molecular contribution to the ISM cross section, only molecular hydrogen is considered. In the gas-phase ISM, the cross section is the sum of the photoionization cross sections of the different elements, weighted by abundance and taking into account depletion onto grains. In addition to the updates to the photoionization cross sections, the gas-phase cross section differs from previous values as a result of updates to the ISM abundances. These updated abundances are available through the `abund wilm` command. Details of updates to the photoionization cross sections as well as to abundances can be found in Wilms, Allen and McCray (2000, ApJ 542, 914). Note that this model differs from tbabs in that grains are not included.

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	redshift

6.3.34 zvarabs

A photoelectric absorption with variable abundances using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. The column for each element is in units of the column in a solar abundance column of an equivalent hydrogen column density of 10^{22} cm^2. The Solar abundance table used is set by the abund command.

`par1–par18`	=	equivalent columns for H, He, C, N, O, Ne, Na, Mg, Al, Si, S, Cl, Ar, Ca, Cr, Fe, Ni, Co
`par19`	=	redshift

6.3.35 zvfeabs

Redshifted photoelectric absorption with all abundances tied to Solar except for iron. The Fe K edge energy is a free parameter.

`par1`	=	equivalent hydrogen column (in units of 10^{22} cm^{-2})
`par2`	=	abundance relative to Solar
`par3`	=	iron abundance relative to Solar
`par4`	=	Fe K edge energy
`par5`	=	Redshift

6.3.36 zvphabs

A redshifted photoelectric absorption with variable abundances using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. The abundances are specified relative to the Solar abundance table set using the abund command. This model is identical to zvarabs except for the way that the parameters are defined.

$$M(E) = \exp(-\texttt{par1}\sigma(E(1+\texttt{par2}))$$

where $\sigma(E)$ is the photo-electric cross-section (NOT including Thomson scattering) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2–par18`	=	abundances for He, C, N, O, Ne, Na, Mg, Al, Si, S, Cl, Ar, Ca, Cr, Fe, Co, Ni wrt to Solar (defined by the abund command)
`par19`	=	redshift

6.3.37 zwabs

A photo-electric absorption using Wisconsin (Morrison and McCammon; ApJ 270, 119) cross-sections.

$$M(E) = \exp(-\texttt{par1}\sigma(E(1+=\texttt{par2})))$$

where σ (E) is the photo-electric cross-section (NOT including Thomson scattering) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	redshift

6.3.38 zwndabs

Photo-electric absorption from approximation to a warm absorber using Balucinska-Church and McCammon (ApJ 400, 699) cross-sections. Relative abundances are set by the `abund` command.

$$\begin{aligned} M(E) &= \exp(-\texttt{par1}\sigma(E(1+\texttt{par3}))) & \text{for } E \geq \texttt{par2} \\ &= 1. & \text{for } E \leq \texttt{par2} \end{aligned}$$

where σ (E) is the photo-electric cross-section (NOT including Thomson scattering) and

`par1`	=	equivalent hydrogen column (in units of 10^{22} atoms/cm^2)
`par2`	=	Window energy (keV)

`par3` = redshift

6.4 Convolution Model Components

6.4.1 gsmooth

Gaussian smoothing with a variable sigma, which varies as the `par2` power of the energy. The sigma at 6 keV is set with `par1`.

$$
\begin{aligned}
dC(E) &= (1./(\sqrt{2\pi\sigma(E)^2}))\exp(-0.5((E-X)/\sigma(E))^2))A(X)dX \\
\sigma(E) &= \texttt{par1}(E/6keV)^{\texttt{par2}}
\end{aligned}
$$

where :

`par1`	=	gaussian sigma at 6 keV
`par2`	=	power of energy for sigma variation

6.4.2 lsmooth

Lorentzian smoothing with a variable width, which varies as the par2 power of the energy. The width at 6 keV is set with par1.

$$
\begin{aligned}
dC(E) &= (\sigma(E)/2\pi)/((E-X)^2+(\sigma(E)/2)^2)A(X)dX \\
\sigma(E) &= \texttt{par1}(E/6keV)^{\texttt{par2}}
\end{aligned}
$$

where :

`par1`	=	lorentzian width at 6 keV
`par2`	=	power of energy for sigma variation

6.4.3 reflect

Convolution model for reflection from neutral material according to the method of Magdziarz & Zdziarski (1995, MNRAS, 273, 837). This is a generalization of the pexrav and bexrav models. When using this model it is essential to extend the energy range over which the model is calculated because photons at higher energies are Compton downscattered into the target energy range. The energy range can be extended using the extend command. The upper limit on the energies should be set above that for which the input spectrum has significant flux. See the help on pexrav or bexrav for further information and admonitions.

`par1`	=	reflection scaling factor (1 for isotropic source above disk)
`par2`	=	redshift, z
`par3`	=	abundance of elements heavier than He relative to the solar abundances
`par4`	=	iron abundance relative to the above
`par5`	=	$\cos i$, the inclination angle

6.4.4 rgsxsrc

Convolution model for the analysis of moderately-extended ($\sim$ 1 arcmin) sources, developed by Andy Rasmussen of the Columbia University XMM-Newton RGS instrument team. The code convolves the spectral model with an angular structure function for a given extended source. The structure function is taken directly from an image (e.g., XMM-Newton EPIC, Chandra ACIS etc) where the user provides RA and Dec (2000) coordinates for the source, position angle of the spacecraft, and an aperture size suitable for the source in order to characterize the convolution function. The model resulting from the convolution is then used with the standard RGS point source spectral response to fit data.

The user is required to have used the XSPEC command `xset` prior to defining the spectral model, e.g.:

`XSPEC> xset rgs_xsource_file <filename>` This command points XSPEC to an external file containing the attitude and aperture information. A typical file must look like this :

RGS_XSOURCE_IMAGE <filename of image of source>

RGS_XSOURCE_BORESIGHT <image boresight in format>

RGS_XSOURCE_EXTRACTION <size of region in arcminutes>

For example :

RGS_XSOURCE_IMAGE </local/data/mymachine/myusername/MOS1.FIT>

RGS_XSOURCE_BORESIGHT <05:25:02.9 -69:38:30 339.760974>

RGS_XSOURCE_EXTRACTION <1.8>

The RA and Dec of the center of the source can be taken determined by the user from the image or taken from the headers of the source spectrum. The position angle can be found in the image headers.

The file is reread on each iteration so editing the file during an XSPEC session will cause these parameters to be changed. The only model parameter is

`par1`	=	the order of the spectrum (this number is always negative).

NB. The interpretation of results using this model is not trivial. The method assumes that the spatial distributions of the continuum and all lines are identical to the broad band image. This is unlikely to be the case. Resulting line velocities and profiles should be treated with appropriate caution.

Contact the US XMM_Newton GOF for help. xmmhelp@athena.gsfc.nasa.gov

6.5 Pile-Up Model Components

6.5.1 pileup

CCD pile-up model used for brightish point sources observed by Chandra. This is an implementation of the fast pile-up algorithm proposed by John Davis (see http://space.mit.edu/%7Edavis/papers.html/pileup2001.ps). The frame time and maximum number of photons to pile up should be fixed. The grade morphing is expressed through a single parameter, alpha, which should be left as a free parameter. This model should be considered in beta test.

`par1`	=	frame time (in seconds)
`par2`	=	maximum number of photons to pile up
`par3`	=	grade correction for single photon detection
`par4`	=	grade morphing parameter (good grade fraction is assumed proportional to par4$^{(p-1)}$ where p is the number of piled photons)
`par5`	=	PSF fraction. Only this fraction will be treated for pile-up
`par6`	=	Number of regions. The counts to be piled-up will be distributed among par6 regions, which will be piled-up independently.

6.6 Mixing Model Components

6.6.1 ascac

Mixing model for ASCA data. Written for cluster data so uses beta or two power-law surface brightness models. Includes a calculation of the telescope effective area so no arf should be applied to input files. Note that this model is very slow if any of the parameters are free.

The model is used by reading spectra in as separate datagroups. Each input file requires the XFLT0001 keyword set to a different number (eg if concentric annuli are in use then number outwards). The normalizations for each datagroup should be linked since the ascac model takes care of the relative normalizations based on the surface brightness model used. A maximum of five different spatial regions is allowed. The absolute normalization is not reliable so this model should not be used to derive fluxes.

`par1`	=	Alpha
`par2`	=	Beta
`par3`	=	Core (arcmin)
`par4`	=	Switch (0 $\Rightarrow$ beta model, 1 $\Rightarrow$ 2-power-law)

6.6.2 projct

This model performs a 3-D to 2-D projection of prolate ellipsoidal shells onto elliptical annuli. The annuli can have varying ellipticities and position angles but must have the same center. The user should extract spectra in a series of annuli. Each spectrum needs three additional keywords (XFLT0001, XFLT0002, XFLT0003) in the spectrum extension. These keywords contain the semi-major axis, semi-minor axis, and position angle (in degrees) for the outer boundary of the annulus. It is assumed that the inner boundary is specified by the outer boundary of the next annulus in. The lengths can be in any consistent units although for numerical accuracy they should have reasonable values. Optional pairs of extra keywords (eg XFLT0004/5, XFLT0006/7, etc.) can be used to specify start and end angles for a partial annulus. These angles should be given relative to the same zero as the position angle.

The user reads in the spectra as separate datagroups and sets model parameters for each datagroup. The model for datagroup J will be the model in the shell whose outer boundary is a prolate ellipsoid of semi-major and semi-minor axes given by the semi-major and semi-minor axes in the XFLT keywords for dataset J. The projct model sums up the appropriate fractions of each ellipsoid model to make the projected spectrum.

For example, suppose we extract spectrum from three elliptical regions defined by (1,0.5,0), (2,1,0), (3,1.5,0). That is the first region is in an ellipse of semi-major axis 1 and semi-minor axis 0.5. The second region is an elliptical annulus whose inner boundary has semi-major axis 1 and semi-minor axes 0.5 and whose outer boundary has semi-major axis 2 and semi-minor axis 1. The third region is defined similarly. The model fit has a temperature of 2 keV for the first datagroup, 3 keV for the second, and 4 keV for the third. The actual model fit to the first dataset has contributions from all three temperatures, the second only from the 3 and 4 keV components, and the third only from the 4 keV component. The weighting is the fraction of the ellipsoidal volume intersected by the elliptical annular cross-section. Thus the normalizations correspond to the emission measure in each ellipsoidal shell.

The projct model has 3 parameters which can be used to define the inner ellipse of the region being analyzed. For instance, in the example above we could have only read in spectra for the outer two regions but then set the projct parameters to (1.0,0.5,0.0). This would have allowed us to determine the temperatures and emission measures of the outer two annuli without having to worry about fitting a model to the central region.

`par1`	=	semi-major axis of inner boundary ellipse
`par2`	=	semi-minor axis of inner boundary ellipse
`par3`	=	position angle of inner boundary ellipse

Chapter 7

Associated programs

7.1 Introduction

A number of programs and subroutine libraries are available for manipulating the files used by XSPEC. Most of these are part of the FTOOLS package of tasks to read, write, and modify FITS-format files. There are a few tasks in the XANADU distribution for manipulating the older pre-FITS file formats. However, we recommend that you use the FTOOLS conversion tasks to turn older files into the FITS format and then use the FTOOLS tasks to manipulate them.

The FTOOLS package is available at ftp from

`ftp:legacy.gsfc.nasa.gov/software/ftools`

or at:

`http://heasarc.gsfc.nasa.gov/docs/software/ftools/ftools_release.html`

If the FTOOLS package is installed, then help is available on any task by typing `fhelp taskname`. To get a complete list of FTOOLS, type `fhelp ftools`.

7.2 FTOOLS reading tasks

FDUMP Prints the contents of a FITS file to the screen or to a file.

DMPRMF Prints the contents of a FITS RMF file to the screen or to a file. This tool prints the RMF file in a more legible fashion than FDUMP.

7.3 FTOOLS manipulation tasks

GRPPHA Defines (or redefines) and/or displays the grouping and quality flags, the important keywords, and the fractional systematic errors.

RBNPHA Compresses a FITS PHA file to a user-defined number of channels. The output is a new file containing the revised PHA extension plus a direct copy of any other extensions in the original file.

RBNRMF Compresses a FITS RMF file (the detector response matrix) in channel space to give a user-defined number of resulting channels.

FPARKEY Changes the value of a keyword in a FITS extension header.

MATHPHA Performs arithmetical operations on PHA files.

ADDARF Adds together ARFs.

ADDRMF Adds together RMFs.

MARFRMF Multiplies an RMF file by an ARF file.

7.4 FTOOLS conversion tasks

SF2PHA Converts an old pre-FITS-format spectral file into a FITS PHA file.

RSP2RMF Converts an old pre-FITS-format response file into a FITS RMF file.

STW2PHA Converts the output of STWFITS (IRAF.stsdas.fitsio) into a FITS PHA file.

7.5 FTOOLS subroutines

The directory **callib/src/gen** contains a number of subroutines for reading and writing the extensions in FITS format spectral and response files. These are summarized below.

RDPHA2	Read a spectrum extension
WTPHA2	Write a spectrum extension
RDRMF4	Read the matrix extension
WTRMF4	Write the matrix extension
RDEBD3	Read the channel boundaries extension
WTEBD3	Write the channel boundaries extension
RDARF1	Read the effective area extension
WTARF1	Write the effective area extension

To use these routines the link line in the makefile should include

–L$FTOOLS/lib –lcaltools –lcftools –lftools –lcfitsio –lxanlib –lreadline –lhistory

Appendix A

Overview of PLT

A.1 Command summary

CLear	Immediately clear the graphics device.
COlor	Change the default colour index.
CONtour	Produce a contour plot.
CPD	Change the plotting device.
CQuit	Clear the graphics device and return control to XSPEC.
CSize	Change the default character size.
Error	Control whether errors are displayed and used in fitting.
EXit	Exit PLT and return control to XSPEC.
Fit	Fit the PLT model to the data.
FNy	Evaluate the model at the specified location.
FOnt	Change the default text font.
Freeze	Freeze a parameter value.
GAp	Change the default gap size between the data and the edge.
Grid	Control the location of the major and minor tic marks.
Hardcopy	Make a file that can later be printed.
HElp	Obtain help on any PLT command.
Imodel	Numerically integrate the model over specified range.
LAbel	Add or remove labels from the plot.
LIne	Control whether a line is used to connect data points.
LOg	Control whether data is plotted using a $\log_{10}$ scale.

LStyle	Change the default style of the line connecting the data points.
LWidth	Change the default line width.
MArker	Control whether the data points are plotted with markers.
MOdel	Define a PLT model.
Newpar	Change a parameter value associated with the model.
Plot	Immediately re-plot the data.
PRompt	Change the "`PLT>`" prompt.
Rescale	Reset the minimum and maximum plot range.
SCr	Change the color representation of the specified color index.
SHow	Display the values of PLT internal variables.
SKip	Control how PLT divides data into vectors.
STatistics	Compute various statistical properties of the data.
THaw	Allow a parameter value to vary during a fit.
Time	Control whether the time stamp is plotted.
Uncertainty	Compute the uncertainty in a parameter value.
VErsion	Display date of the most recent modification to PLT.
Viewport	Control the size of the viewport plotting area.
WData	Write a QDP data file to disk.
WEnviron	Write both QDP data and header files to disk.
WHead	Write a QDP header file to disk.
WModel	Write a model file to disk.
Xaxis	Define the method used to calculate the x-variable.
Yaxis	Define the y-axis scale for a contour plot.
$	Execute operating system commands.
@⟨filename⟩	Read commands from a PLT command (.PCO) file.

A.2 XSPEC graphics

Extensive documentation for the PLT graphics routine is available in the "The QDP/PLT Users's Guide" and also from the interactive help. This appendix is intended to provide information to assist in using PLT from within the XSPEC program.

Within XSPEC, it is possible to set your graphics device using the `cpd` command. Any PGPLOT[1] device supported by your local version of PGPLOT is accepted. The `cpd` command can be used to display a list of all PGPLOT devices. If you fail to enter a device name, you will be prompted for a PGPLOT device every time you generate a new plot.

From XSPEC, there are two ways to call the PLT routine. The `plot` command will produce a graph and control will return immediately to XSPEC. To go into interactive plot mode, use `iplot`. This will produce exactly the same graph but the "`PLT>`" prompt will appear. At this point, you can enter PLT commands to inspect interesting parts of the graph, add labels, or make a hardcopy file for later printing.

A.3 Getting started with PLT

Typing `HElp` at the "`PLT>`" prompt will provide you with help concerning the PLT commands. The PLT interactive help is used in exactly the same way as the XSPEC help. If you wish to read the interactive help on PLT without running the PLT routine, you should use the (system level) XHELP PLT command.

The most common command used in PLT is `Rescale`. Using `Rescale X` followed by two numbers, will set the minimum and maximum of the plotted x-range to the numbers specified. If you type `Rescale X` (with no arguments), the minimum and maximum values will be reset to their default values. Likewise, `Rescale Y` can be used to set the y-range.

Most PLT commands do not cause the screen to be updated the way the `Rescale` command does, thus allowing you to make several changes to the the graph without having to wait for the screen to be updated after every change. If, at any time, you wish to see what the current graph looks like, enter the `Plot` command, which will cause the display to be redrawn.

The `LAbel` command can be used to add labels to various locations on the graph. For example, typing `LAbel Top EXOSAT was great` will cause the message 'EXOSAT was great' to appear at the top of the graph the next time the display is redrawn. The command `LAbel Top` followed by a return will remove that message.

The `Hardcopy` command is used create a file that can later be printed. `Hardcopy` does not reproduce what currently is visible on the graphics display, but rather what you would see if you re-issued the `Plot` command. The command `Hardcopy ?` can be used to see what the default hardcopy device is. If you do not like the default, you can override it. For example, assume that your version of PGPLOT supports the QMS device. To generate a file suitable for printing on a QMS printer you would enter `Hardcopy /QMS` and a `PGPLOT.QMPLOT` file will be generated in your current directory. Many sites have created a system-wide indirect file that will both create the hardcopy file, and then immediately print it. If you are at one of these sites, then `@HARD` will directly produce a plot on the printer.

In PLT it is possible to fit models to the displayed data. However, it is important to remember that PLT does not fold the model through the detector response, and therefore, PLT models should not be used to fit X-ray spectra.

To exit PLT and return to the XSPEC level, enter the PLT `EXit` command. When you return to XSPEC, any changes that you made to the plot will be forgotten.

[1] PGPLOT is the name of a Graphics Subroutine Library written by T. J. Pearson at the California Institute of Technology.

Appendix B

Fitting with few counts/bin

B.1 Theory

B.1.1 No background

Cash (ApJ **228**, 939) showed that the χ^2minimization criterion is a very bad one if any of the observed data bins had few counts. A better criterion is to use a likelihood function :

$$C = 2\sum_{i=1}^{N}(y(x_i) - y_i \ln y(x_i) + \ln y_i!)$$

where y_i are the observed data and $y(x_i)$ the values of the function. Minimizing C for some model gives the best-fit parameters. Furthermore, this statistic can be used in the same, familiar way as the χ^2statistic to find confidence intervals. One finds the parameter values that give $C = C_{min} + N$, where N is the same number that gives the required confidence for the number of interesting parameters as for the χ^2case.

Castor (priv. comm.) has pointed out that a better function to use is :

$$C = 2\sum_{i=1}^{N}(y(x_i) - y_i + y_i(\ln y_i - \ln y(x_i))$$

This differs from the first function by a quantity that depends only upon the data. In the limit of a large number of counts this second function does provide a goodness-of-fit criterion similar to that of $\chi^2$and it is now used in XSPEC. It is important to note that the C-statistic assumes that the error on the counts is pure Poisson, and thus it cannot deal with data that already has been background subtracted, or has systematic errors.

B.1.2 With background

Arnaud (2001, ApJ submitted) has extended the method of Cash to include the case when a background spectrum is also in use. Note that this requires the source and background spectra to both be available, it does not work on a background-subtracted spectrum.

Suppose we have an observation which produces S_i events in the $i = \{1, , N\}$ spectral bins in an exposure time of t_s. This observation includes events from the source of interest along with background events. Further suppose that we perform a background observation which generates B_i events in an exposure time t_b. If the source count rate in bin i is y_i then the new fit statistic is

$$W = 2\sum\{t_s y_i + (t_s + t_b)f_i - S_i \log(t_s y_i + t_s f_i) - B_i \log(t_b f_i) - S_i(1 - \log S_i) - B_i(1 - \log B_i)\}$$

where

$$f_i = \frac{S_i + B_i - (t_s + t_b)y_i + d_i}{2(t_s + t_b)}$$

and

$$d_i = \sqrt{[(t_s + t_b)m_i - S_i - B_i]^2 + 4(t_s + t_b)B_i m_i}$$

In the limit of large numbers of counts/bin a second-order Taylor expansion shows that W tends to

$$\sum \frac{[S_i - t_s y_i - t_s f_i]^2}{t_s y_i + t_s f_i} + \frac{[B_i - t_b f_i]^2}{t_b f_i}$$

which is distributed as χ^2 with $(N - M)$ degrees of freedom, where the model y_i has M parameters (including the normalization).

B.2 Practice

B.2.1 No background

XSPEC uses a variant of Marquardt's algorithm described in §11.5 of "Data Reduction and Error Analysis for the Physical Sciences" by Bevington. (The reader is advised that this description is designed to be read in conjunction with Bevington.) The algorithm turns on finding a matrix α_{jk} and a vector β_k such that the equation :

$$\beta_k = \sum_j \delta a_j \alpha_{jk}$$

gives sensible values of the change in parameters, δa_j, for the fitting function. Bevington §11.4 gives the derivation of α and β and shows that β is parallel to the gradient of χ^2.

Now the C statistic has a gradient with respect to the parameters of the fitting function of :

$$-\frac{1}{2}\frac{\partial C}{\partial a_k} = \sum_i (\frac{y_i}{y} - 1)\frac{\partial y}{\partial a_k} = \beta_k$$

So, following Bevington, expand $y(x_i)$ about y_0 :

$$y(x_i) = y_0(x_i) + \sum_j \frac{\partial y_0(x_i)}{\partial a_j}\delta a_j$$

substitute into C and minimize with respect to the changes in the parameters :

$$\frac{\partial C}{\partial \delta a_k} = 2\sum_i (\frac{\partial y_0(x_i)}{\partial a_k} - y_i(y_0(x_i) + \sum_j \frac{\partial y_0(x_i)}{\partial a_j}\delta a_j)^{-1}\frac{\partial y_0(x_i)}{\partial a_k}) = 0$$

so to first order in the parameter changes :

$$\sum_i (\frac{y_i}{y_0(x_i)} - 1)\frac{\partial y_0(x_i)}{\partial a_k} = \sum_j (\sum_i \frac{y_i}{y_0^2(x_i)}\frac{\partial y_0(x_i)}{\partial a_j}\frac{\partial y_0(x_i)}{\partial a_k}\delta a_j)$$

or :

$$\beta_k = \sum_j \delta a_j \alpha_{jk}$$

where :

$$\alpha_{jk} = \sum_i \frac{y_i}{y_0^2(x_i)} \frac{\partial y_0(x_i)}{\partial a_j} \frac{\partial y_0(x_i)}{\partial a_k}$$

These α and β then are substituted for those used in the χ^2case and the algorithm works as required. Note that α_{jk} is $-(\frac{\partial^2 C}{\partial a_k \partial a_j})$ to first order in partial derivatives in y, evaluated at y_0.

There is one further difference in XSPEC between the $\chi^2$and likelihood methods, which is caused by the fact that XSPEC uses an analytic formula for setting the model normalisation. In the χ^2case, this means multiplying the current model by :

$$(\sum_i \frac{y_i y(x_i)}{\sigma_i^2})/(\sum_i \frac{y(x_i)^2}{\sigma_i^2})$$

where σ_i is the error on y_i. In the likelihood case the corresponding factor is :

$$\sum_i y_i / \sum_i y(x_i)$$

B.2.2 With background

An analogous argument to the above can be followed through for the W statistic. We need the partial derivatives of W which are evaluated as follows.

$$\frac{\partial W}{\partial a_j} = 2 \sum \left\{ t_s + g_i(t_s + t_b) - \frac{N_i(1+g_i)}{y_i + f_i} - \frac{B_i g_i}{f_i} \right\} \frac{\partial y_i}{\partial \alpha_j}$$

$$\frac{\partial^2 W}{\partial a_j \partial a_k} = 2 \sum \left\{ (t_s + t_b) h_i - \frac{N_i h_i}{y_i + f_i} + \frac{N_i(1+g_i)^2}{(y_i+f_i)^2} - \frac{B_i h_i}{f_i} + \frac{B_i g_i^2}{f_i^2} \right\} \frac{\partial y_i}{\partial \alpha_j} \frac{\partial y_i}{\partial \alpha_k}$$

where

$$g_i \frac{\partial y_i}{\partial \alpha_j} = \frac{\partial f_i}{\partial \alpha_j} = \frac{1}{2d_i}((t_s + t_b) y_i - N_i + B_i - d_i) \frac{\partial y_i}{\partial \alpha_j}$$

$$h_i \frac{\partial y_i}{\partial \alpha_k} = \frac{\partial g_i}{\partial \alpha_k} = \frac{2(t_s + t_b) N_i B_i}{d_i^3} \frac{\partial y_i}{\partial \alpha_k}$$

Note that in this case there is no analytic formula that can be used to set the model normalization.

Appendix C

Adding models to XSPEC

C.1 Analytic Models

If you have a model that will be used frequently, you may want to include it among the standard models so that it will be listed after the `model ?` command. To do so, you first must create a subroutine that calculates the model spectrum given an input array of energy bins and an array of parameter values. The spectrum for an **additive** model should be in terms of photons/cm^2/s (not photons/cm^2/s/keV), while for a **multiplicative** model it is the multiplicative factor for that bin.

The six arguments in the calling sequence are EAR(0:NE), NE, PARAM(*), IFL, PHOTAR(NE), and PHOTER(NE) . EAR(0:NE) are input and are the boundaries for the NE energy bins. These are set by the response matrices of the detectors in use, so you should not make any assumptions about the values of EAR. PARAM(*) are input and are the values of the model parameters. IFL is an integer which specifies which dataset these energies are for, it exists to allow multi-dimensional models where the function might also depend on eg pulse-phase in a variable source. PHOTAR(NE) is the output array and should not be assumed to have any particular values on input. PHOTER is an array that allows the function to return model variances.

To set the default parameters and tell XSPEC that your model exists, you must add to the file `lmodel.dat` in a subdirectory specified by the environment variable $LMODDIR. As an example, suppose the model `newmodel` with 3 parameters is implemented with the fortran 77 subroutine newfunc with

```
SUBROUTINE NEWFUNC(EAR,NE,PARAM,IFL,PHOTAR,PHOTER)
INTEGER IFL, NE
REAL*4 EAR(0:NE), PARAM(*), PHOTAR(NE), PHOTER(NE)
...

END
```

The corresponding lmodel.dat entry is:

```
newmodel        3  0.          1.e20           newfunc    add  0
kT      keV     1.    0.008   0.008   64.0      64.0      .01
Abundanc " "    1.    0.      0.      5.        5.        -0.001
Redshift " "    0.    0.      0.      2.        2.        -0.001
```

The first line for each model gives the model name, the number of parameters, the low and high energies for which the model is valid, the name of the subroutine in the **functions** subdirectory, and the type of model (add, mul, mix, or con). The final argument is a flag which should be set to 1 if PHOTER is calculated by NEWFUNC.

There then should be one succeeding line for each parameter. Each line contains the parameter name, a default starting value, the hard minimum, the soft minimum, the soft maximum, the hard maximum, and the initial delta.

C.1.1 Linking User-Defined Analytic models to XSPEC

The object of the updated local model implementation in XSPEC V11 is to remove the need for users to build personal copies of XSPEC. Instead, the user should be able to define a directory that contains their models, build a library from it that contains the functions XSPEC uses to recognize valid model components, and link the system-wide version of XSPEC to that library by setting appropriate environment variables.

At Installation Time

To build and install local models directly into a full Xspec build, use the following procedure. Prior to starting the build, type (csh, tcsh):

```
setenv LMODDIR /path/to/your/local/model/code
```

or

```
$LMODDIR=/path/to/your/local/model/code
export LMODDIR
```

in Bourne-type shells (sh, bash, ksh). Then proceed with the standard build instructions. This will create a library in the directory $LMODDIR that XSPEC will link to at runtime. Note that you must have $LMODDIR defined whenever you run XSPEC with your local models.

Using your own models in a site-installed version of XSPEC

To build and use local models in a private directory, after XSPEC has been installed perform the standard LHEASOFT setup:

```
setenv LHEASOFT /path/to/installed/xspec/ending/in/ARCHDIR
cd $LHEASOFT/BUILD_DIR
./configure
source $LHEASOFT/lhea-init.csh
```

(and similarly for Bourne shell and variants), where ARCHDIR is the architecture-specific directory containing the installed software (e.g. Linux_2.2_i686 for a Pentium-II based PC running the Linux 2.2 kernel, as supplied with RedHat Linux 6.x, or SunOS_5.6_sparc for Solaris 2.6). In addition, perform the extra "developers" setup:

```
source $LHEASOFT/BUILD_DIR/devinit.csh
```

this will put the `hmake` script in your path. Now, to build your local models, type:

```
setenv LMODDIR /path/to/your/local/models/code
```

then cd to the directory:

```
$LHEASOFT/../src/spectral/xspec/src/local_mod
```

and type “hmake”. This should build your local models in the $LMODDIR directory. N.B. Files named `Makefile` and `setup.dat` will be copied into the directory given by $LMODDIR, so if you have files with those names, please back them up. Also, note that if there are any site-wide installed local models, you will not be able to use these unless you copy their code and lmodel.dat entries into your $LMODDIR directory.

If all goes well, at the conclusion of this step, $LMODDIR will contain a file called `libxspec_lfn.so`, which is a shared library containing the local model code. Finally add $LMODDIR to the LD_LIBRARY_PATH environment variable. For C shell variants, type:

```
setenv LD_LIBRARY_PATH "$LMODDIR:$LD_LIBRARY_PATH"
```

or for Bourne shell variants:

```
LD_LIBRARY_PATH="$LMODDIR:$LD_LIBRARY_PATH"
export LD_LIBRARY_PATH
```

This step must be performed manually by the user, but may be added to the .login/.profile file. Note that you must have $MODDIR defined whenever you run XSPEC with your local models.

C.2 Table models

A very simple way of fitting with user-defined models is available for a particular class of models. These are models that can be defined by a grid of spectra, with the elements of the grid covering the range of values of the parameters of the model. For instance, for a one-parameter model, a set of model spectra can be tabulated for different values of the parameter (P1, P2, P3, etc.) The correct model spectra for a value P then is calculated by interpolation on the grid. The generalisation to more parameters works in the obvious way. As with standard models, the spectra should be in terms of flux-per-bin and not flux-per-keV. Any set of energy bins can be used, and XSPEC will interpolate the model spectra onto the appropriate energy bins for the detectors in use. It is therefore a good idea to choose energy bins such that the spectrum is well-sampled over the range of interest. The file structure for these models is a FITS format and is described in the OGIP memo OGIP/92-009[1], also available by anonymous ftp[2].

[1] `http://heasarc.gsfc.nasa.gov/docs/heasarc/ofwg/docs/summary/ogip_92_009_summary.html`
[2] `ftp://legacy.gsfc.nasa.gov/fits_info/fits_formats/docs/general/ogip_92_009`

Appendix D

The User Interface

D.1 Introduction

All communication with the user in XSPEC is performed through the XSPEC parser, which itself sits on top of the tcl user interface. The tcl interface identifies the commands being given and provides a number of facilities including command recall, looping, and contingent control. The arguments to commands are interpreted by the XSPEC parser which extracts ranges, switches, and so on. This appendix summarizes the features of these parsers from the point of view of the user.

D.2 XSPEC and tcl/tk

XSPEC uses the Tool Command Language (tcl) to process command input. Because tcl is a full scripting language, this interface allows users to write complex scripts with loops, branching, etc., which utilize XSPEC commands. In this section of the help file we describe how to use those features of tcl necessary to give the user similar functionality to that available in previous versions of XSPEC, and to give information on the details of our tcl implementation that may be useful to experienced tcl users. For a description of tcl, see, for example, ”Practical Programming in Tcl and Tk”, B. Welch, (1997, Prentice Hall).

Tk, tcl’s companion graphical user interface (GUI) toolkit, is also loaded by XSPEC on startup. It is planned that future versions of XSPEC will provide an optional GUI side-by-side with the command line interface (CLI). Although XSPEC does not currently use tk, its presence allows users to write XSPEC scripts with graphical interfaces using Tk commands.

D.2.1 A note on command processing

XSPEC’s older command parser, described in D.4, is still used, and differs from tcl in a number of ways. As a result, the version of tcl that process XSPEC commands must understand features of the XSPEC parser. In tcl, commands and their arguments are delimited by white space. They are terminated by a newline or semicolon, unless there is an open set of parentheses ‘{ }’ constituting a loop or test structure (i.e. while, for or if). XSPEC differs from tcl by treating xspec commands as having a single argument (alphanumeric characters, white space, punctuation, and all), which is then passed to XSPEC’s internal routines intact.

The # sign is used for comments in tcl, but may appear only at the beginning of a command. tcl and XSPEC both ignore carriage returns on a new line, but XSPEC also ignores the infinite skip character,‘/’. The \ character is used in tcl for continuing a command onto the next line. The old - continuation character from previous versions of XSPEC is also supported, but may be deprecated in future releases.

D.2.2 Command Recall/Editing

The XSPEC/tcl interface also uses gnu readline for command input, which allows command line editing and interactive command recall. On most systems, the left and right arrow keys and the backspace/delete key can be used to navigate and edit the command line. The up and down arrow keys can be used to step thru the command history list. Gnu readline is highly customizable, and many more editing/recall functions are available. Readline documentation can be generated in either postscript or html format from the files in the xanadu/readline/doc directory distributed with the source.

The gnu readline inteface can be turned on or off as required, using the command `readline (on|off)`. Giving the command without any arguments displays the current state of the readline interface (enabled or disable). The default is for readline to be enabled.

The default implementation of tcl also supports a C-shell like command recall mechanism. The `history` command gives a numbered list of the most recently entered commands. Any command in the list can be re-executed by entering `!n`, where `n` is the number of the command in the history list. The previous command can be re-executed by entering `!!`. The most recent command that begins with a string can be re-executed by entering `!prefix`, where `prefix` is the string the command begins with.

Note that command recall is implemented using the tcl `unknown` procedure, part of which is a script file loaded by tcl at run time and may be different or not exist on your system. See the section in this help file on the `unknown` command for more details on how it is implemented in XSPEC.

D.2.3 Logging

The `log` command can be used to open a log file to which all input and and output to tcl will be written. Reading these log files can potentially be confusing when logging tcl flow control commands such as `while` or `for`. This is because tcl treats the body of these commands as an argument of the command. Thus when the command is echoed to the log file, the entire body of the command is echoed with it.

In order to make this situation less confusing, before commands are echoed to the command file, all newline characters are replaced by semicolons, and the resulting command line is trucated to 80 characters. Then any commands executed with in the body of a flow control command are echoed as they are executed.

Consider the following sequence of tcl commands within XSPEC:

```
XSPEC> log
Logging to file: xspec.log
XSPEC> set i 1 ; set product 1
1
XSPEC> while {$i <= 5} {
XSPEC> set product [expr $product * $i]
XSPEC> incr i
XSPEC> }
XSPEC> set product
120
XSPEC>
```

This would produce the following output in the file `xspec.log`:

```
Logging to file: xspec.log
XSPEC> set i 1
set product 1
1
XSPEC> while {$i <= 5} {;set product [expr $product * $i];incr i;}
  expr $product * $i
```

```
  set product [expr $product * $i]
  incr i
   expr $product * $i
  set product [expr $product * $i]
  incr i
   expr $product * $i
  set product [expr $product * $i]
  incr i
   expr $product * $i
  set product [expr $product * $i]
  incr i
   expr $product * $i
  set product [expr $product * $i]
  incr i
XSPEC> set product
120
XSPEC>
```

D.2.4 Command Completion

tcl attempts to match the name of any entered command as an abbreviation of a valid command (either a tcl or XSPEC command). If the entered command matches more than one valid command, tcl then lists the possible choices, but does not execute the command. For XSPEC commands, aliases have been constructed matching the command to its minimum abbreviation, as listed when typing '?' at the XSPEC prompt (see aliases). For example, the minimum abbreviation for the 'plot' command is 'pl'. Thus, typing 'pl' will execute the plot command, even though this would otherwise be ambiguous with the tk command 'place'.

Command completion is also implemented using the tcl `unknown` procedure, part of which is a script file loaded by tcl at run time, and may be different or not exist on your system. See the section in this help file on the `unknown` command for more details on how it is implemented in XSPEC.

N.B. tcl explicitly switches off command completion for scripts. Because of the way scripts are implemented in XSPEC, however, command abbreviations nevertheless do work in scripts entered with the `@` command, but not when entered from the command line or using the source command. See below for more details about tcl scripting.

D.2.5 Unix Shell Commands

Shell commands can be executed within XSPEC using the `exec` command (see the help entry on the `exec` command). When running interactively, if tcl cannot find a command which matches that entered on the command line, it will search for a shell command which matches the entered command. If it finds a match, it automatically executes the shell command via `exec`. Note that this feature is implemented using the tcl `unknown` procedure, part of which is a script file loaded by tcl at run time, and may be different or not exist on your system. See the section in this help file on the `unknown` command for more details on how it is implemented in XSPEC.

Note that the tcl `exec` command executes the given command directly, without first passing it on to the shell. Thus no globbing (ie. expansion of wildcards such `*.pha`) is performed. If you wish to pass you command through a shell for wildcard expansion, etc, use the `syscall` command.

If you want to start a subshell from within XSPEC, simple type the command for starting that shell, ie. type `csh` in order to start a C-shell. Note that typing `exec csh` will not work properly (the full expanded command name necessary in this case would be `exec >&@stdout <@stdin csh`). Giving the `syscall` command with no arguments will automatically start a subshell using your current shell.

D.2.6 `unknown` Procedure

tcl provides a facility whereby if it cannot match an entered command to its list of known commands, it calls the `unknown` procedure, with the unmatched command (along with its arguments) as its argument. The version of `init.tcl` distributed with tcl contains a version of the unknown procedure. When tcl initializes, it looks in several standard places for a script file named `init.tcl`, which it executes if found. The `unknown` procedure is where tcl does command completion and automatic shell command execution.

XSPEC has its own special version of the `unknown` procedure, which it uses to intercept script processing requests of the form `@<script>`. At start up time, XSPEC loads its own `unknown` procedure, and renames the previously defined `unknown` procedure to `tclunknown`. If XSPEC is not doing any special processing, it simply passes any unmatched commands on to `tclunknown`, which then processes them as usual.

These factors need to be taken into consideration for programmers writing tcl scripts for use within XSPEC. For example, if after initialization, users wishing to load a different version of the standard tcl `unknown` procedure should name that procedure `tclunknown`, rather than `unknown`.

D.2.7 Aliases

Command name aliases can be constructed using the tcl `interp` command:

```
interp alias {} <command_alias> {} <xspec_command>
```

where `<xspec_command>` is the name of the command you wish to make an alias for, and `<command_alias>` is the name of the alias you wish to set for the command. The {} are required syntax.

To delete the alias `<command_alias>` use the following:

```
interp alias {} <command_alias> {}
```

D.2.8 Initialization Script

When running interactively, the user has the option of providing an initialization script, which will be executed after XSPEC completes its startup procedure, ie. just before it begins prompting for commands. The file should be named `.xspecrc` and located in the user's home directory. If one specifies on the command line a script to run with XSPEC, the initialization script is **not** executed.

D.2.9 XSPEC Command Result

After being executed, many tcl commands return a result string, which is echoed to the terminal when the command is entered on the command line. When writing complex tcl scripts, this result can be stored and/or used as a test in loops, etc. When XSPEC commands are executed, they write information to the terminal by writing directly to the appropriate output channel. However, when running interactively, the tcl result string is also written to the terminal after the command is executed.

Whether or not XSPEC specific commands actually return tcl results is controlled by the tcl variable `xs_return_result`. If this variable is non-zero, XSPEC specific commands will return tcl results. The default value is 0 (no tcl results returned).

Essential all XSPEC commands which print the current value of the fit statistic also return this fit statistic as their tcl result. Other commands which return tcl results are `freeze`, `flux`, `eqwidth`, `lumin`, `error`, and `thleqw`. The `show` command returns various different tcl results based on its options. See the help item for the `show` command for more details.

D.3 Script Files

XSPEC/tcl script files can be executed in three different ways, as follows:

`%xspec - <script>`	!	executing script on initialization
`XSPEC>@<script>`	!	executing script from within the program
`XSPEC>source <tclscript>`	!	use tcl's source command from within the program

Each of these usages does something slightly different. In the first form, XSPEC will execute a file called <script>. One may execute a series of script files at startup with the following command syntax:

```
unix> xspec - file1 file2 file3 ...
```

Note that the space following the `-` is required.

The second form is `@<name>`, where `<name>` is the name of the script file to be executed. Here the default extension of `.xcm` is assumed. Scripts containing valid tcl or XSPEC commands will be executed using this form, and (unless the script ends in "quit" or "exit") will return to the interactive prompt after completion.

The final form, using tcl's source command, is intended for the special case where the script contains the implementation of a new command written in tcl/tk. It will *not* work for general scripts containing XSPEC/tcl commands, for example those produced by XSPEC's `save` command. These should rather be executed using the `@` form.

Note that only in the second case `@` is there a default filename suffix: for both the other methods of script execution the filename must be given in full.

As mentioned above, command abbreviations can be used in scripts executed using the `@` syntax, to allow backward compatibility with scripts written for XSPEC 10, but not otherwise. Therefore, it is recommended that users choose avoid the use of command abbreviations in all scripts to avoid suprise failures when, for example, entering a script name from the command line.

Command Echoing By default, when XSPEC is executing a script file, it echoes each command to the terminal before it is executed. This can be controlled using the tcl variable `xs_echo_script`, whose default value is 1. If this variable is set to 0, the commands from the script file will not be echoed to the terminal.

Converting scripts from XSPEC V.9 and Earlier

Given the differences in command syntax between the current version of XSPEC and that of versions 9.02 and earlier, a perl script file has been produced to aid in converting script files from XSPEC version 9.02 and eariler into a format that will run under the current version of XSPEC. This script is called `xs_update.pl`, and can be found in the directory `$XANADU/spectral/xspec/src/tools`. See the `Script_conversion` help topic in the XSPEC help facility for details on using this script.

D.3.1 Usage Advice

We suggest the following convention:

1. Running an xspec script from the unix command prompt is intended to be used for background processing or overnight batch jobs. Using the unix 'at' command, one can arrange to receive the log file by e-mail.

2. The @ usage is intended for processing previously run xspec command sequences, such as are produced by the `save` command.

3. The `source` usage, as well as executing the commands in the script, performs the equivalent of pre-compiling the script for later invocation. Its most appropriate use is in preparing new custom XSPEC command procedures. Once the script is working correctly, it can be placed in the $XSPEC_HOME directory and become part of the user's standard command set. For examples, see the implementation of the `addline` and `modid` commands, which is through scripts to be found in the directory `src/xanadu/spectral/xspec/manager`. These also show how to make commands self-documenting.

Writing Custom XSPEC commands

XSPEC commands can be written by users as tcl procedures, which have similarities with fortran subroutines. Within XSPEC, tcl procedures can take arguments and execute XSPEC and tcl commands. The syntax for specifying arguments to a tcl procedure is as follows:

```
proc my_proc {arg1 arg2}{

...

data 1:1 ${arg1}_s0_20
data 2:2 ${arg2}_s1_20
...

}
```

Here, arg1, arg2 are values supplied by the user (here, part of a filename) from the command line, and substituted wherever `${arg1}, ${arg2}` appear within the script. One may also give an argument a default value, so that the command so created may be invoked even without needing to specify the argument:

```
proc my_proc {arg1    {arg2 file2} }  {
...

}
```

Note that the parentheses enclosing both `arg2` and `file2` in this expression distinguish this from the case where 3 arguments are required for `my_proc`. Once this file is created, it needs to be `source`'d once, which compiles the script into an internal bytecode representation (this is similar to the way Java operates). Alternatively, one may place it in the $XSPEC_HOME directory before starting XSPEC, in which case it will be found automatically and compiled the first time it is invoked. The `my_proc` procedure is then defined such that one may type:

```
XSPEC>my_proc eso103 eso104
```

And then the data statement in the above example will be executed as if the following had been entered:

```
data 1:1 eso103_s0_20
data 2:2 eso104_s1_20
```

The tcl `info` command can be used to show which procedures have been defined:

```
XSPEC>info commands <procedure name>
```

This will return <procedure name> if that procedure has been compiled already or is a built-in command, or nothing if it has not (yet) been invoked or defined.

Scripting commands that prompt the user

The commands `model`, `editmod`, `addmod`, `newpar`, and `fakeit` may prompt the user for more information when used interactively. In order to write scripts that use these commands, one must know how to force XSPEC to enter the information that would be prompted for. The technique is exemplified as follows. Suppose we defined a procedure `xmodel` that makes a model with certain predefined parameter values:

```
set p1  {1.5 0.001 0 0 1.E05 1.E06}
set p2  {1 0.001 0 0 1.E05 1.E06}


 proc xmodel {modelString param1  param2  args} {

   ...

  model $modelString & $param1 & param2 & /*

        ...
 }
```

In this context, the & character is taken by XSPEC as a carriage return, delimiting the model string and parameter arguments into separate input lines.

The procedure xmodel may be compiled with the command

```
XSPEC> source xmodel.tcl
```

This creates xmodel as a command with two arguments which sets subsequent parameters to their default values. It can be invoked e.g. by

```
XSPEC>xmodel {wa(po + peg)} $p1 $p2
```

Note that the model string, which contains spaces, needs to be entered in { } or quotes. Note also the "args" argument, (not used here) that tcl uses to supply a variable number of arguments to a procedure (it is supplied as a tcl list, which can be split within the procedure into separate strings for digestion by xspec if present).

D.3.2 tcl Script Example

In the directory `$XANADU/spectral/session` is a script file called `tclex.xcm`. This script gives an example of how one might use the power of tcl's scripting language in an XSPEC session.

This script should be executed with

```
XSPEC> @tclex

# This script gives an example of how one might use the power of tcl's
# scripting language in an XSPEC session.  In this example, XSPEC loops
# thru 3 data files (file1, file2 and file3) and fits them each to the
# same model 'wabs(po+ga)'.  After the fit the value of parameter 4 (the
# line energy for the gaussian) for each data set is saved to a file.
```

```
# Keep going until fit converges.
query yes

# Open the file to put the results in.
set fileid [open fit_result.dat w]

for {set i 1} {$i < 4} {incr i} {

# Set up the model.
  model wabs(po+ga) & /*

# Get the file.
  data file$i

# Fit it to the model.
  fit

# Get the values specified for parameter 4.
  tclout param 4
  set par4 [string trim $xspec_tclout]

# Turn it into a Tcl list.
  regsub -all { +} $par4 { } cpar4
  set lpar4 [split $cpar4]

# Print out the result to the file.  Parameter value is
# the 0th element of the list ‘lpar4’.
  puts $fileid "$i [lindex $lpar4 0]"

}

# Close the file.
close $fileid
```

The user is encouraged to read the voluminous on-line documentation and literature available about tcl in order to benefit fully its flexible command processing, graphical interfacing, and scripting capabilities. See the Scriptics WWW site[1] for much more information and extensive bibliography.

D.4 The XSPEC Parser

D.4.1 Basics

The parser routines provide a general, easy interface between a program and the user, particularly for command-driven or other highly interactive programs. The basic cycle begins with the user inputing a line (optionally after some prompt) on the terminal. This line then can be analyzed by the program as a string of arguments. The program then can try to interpret each argument as a number or as a character string. Character strings can be checked if they are one of an allowed list of options. The actual details of how a line is interpreted depend on the particular program and how it uses the various parser facilities. An example of an input line might be:

```
Foo , 4.7 "Some information" - ! A comment
```

[1] http://www.scriptics.com

which is divided into three arguments: ‘Foo’, ‘4.7’ (which can be interpreted as a number), and “Some information.”

When parsing an input line, there are several special characters:

`blank`	( )	(an horizontal tab is also a blank)
`comma`	(,)	
`special delim.`	(|)	(the vertical bar)
`comment char.`	(!)	
`begin quote`	(”)	(Note, the begin and end quote are the
`end quote`	(”)	same character by default, the ”)

There are special characters whose interpretation depends on their position in the line:

`"infinite skip"`	`/`
`continuation`	`-`
`end of file`	`/*`

Delimiting arguments

When parsing a line for an argument, initial and trailing blanks are ignored. In addition, an argument field is delimited by a comma, or some of the special characters (particularly the comment, end of input, special delimiter characters). If an argument has no non-blank characters before a delimiter is reached, the argument is said to be an empty, null, or skipped argument. Conventionally, an empty argument is to be treated as indicating some default value. For example:

`foo`	! is a line consisting of a single argument ‘foo’.
`foo bar`	
`foo,bar`	
`foo   bar`	
`foo,  bar`	
`foo  , bar`	! are all lines with two arguments ‘foo’ followed
`foo|bar`	! by ‘bar’.
`foo`	! have two initial empty arguments followed by
`foo`	! ‘foo’.
`bar,,foo`	! has ‘bar’, an empty argument and ‘foo’.

Comments

When looking for arguments, the comment character is treated logically as the end of the line. Thus, any following characters are ignored (and can be used as a comment). Thus, the line

`foo !  bar`	! has a single argument ‘foo’ followed by a comment.
	! Comment strings cannot be read by the program.
`,`	!a comment ! has two empty arguments before the comment.

Quotes

If a user wishes to input an argument that contains a blank, comma, comment character, special delimiter, etc., then the argument should begin with the begin quote (") character, and have an end quote (") character at the end. Reaching the end of the input line is equivalent to having the end quote character. If you want to have the string include the end quote character, then you should double the character (similar to the way that FORTRAN handles character strings with embedded single quotes). By default, the begin quote and end quote characters are the same (the ASCII double-quote: "). For example:

```
"This argument contains blanks"
"This argument ends with a single """," followed by a 2nd argument"
"The initial "" allow the words following the ! to be read"
" A final "" is optional for the last argument
```

The infinite skip character: /

The function of some characters depends on their location in the input line. For example, if the infinite skip character (/) is the last non-blank character on the line, or the last before the comment character, it is treated as a character delimiter, similar to a comma. In addition, it is equivalent to an infinite series of skipped or empty arguments existing before the end-of-line condition is raised. This treatment is mostly for compatibility with the standard FORTRAN List Directed I/O standards where a slash (/) indicates that all remaining arguments of a read are to retain their current values. If the character is placed anywhere else in the line, it is just an ordinary character and will be treated as part of an argument. Examples of lines that use the infinite skip character:

Foo, bar, bletch	/
Foo bar bletch,/	
Foo bar bletch/	! a comment after the infinite skip char

all have three non-empty arguments, followed by infinite skips. In the following cases the slash is NOT treated as an infinite skip character:

file/device	! This is a single argument ‘file/device’
foo/,	! This is a single argument ‘foo/’
”foo/”	! As is this.

The continuation character: -

If the continuation character (-) is the last non-blank character on a line, or the last before the comment character, then the next line of input will be treated as logically concatenated to the current line. For terminal input, the user will receive a prompt “->” indicating that the line is treated as a continuation. Note that the continuation character is NOT a delimiter, and thus must be separated from any earlier argument by a blank or some other delimiter. Examples:

```
These two lines -
are really one line !with 7 arguments
This line is not concatenated- !the 5th argument is "concatenated-"
```

```
with the next.
Even if you concatenate -!you are allowed to have comments
on each input line ! this was equivalent to an 8 argument line.
```

Because of the way continuations are processed, the user may not see the prompt until the program processes the last argument on the line. Thus, the type-ahead for the continuation may be ‘invisible’.

The terminator: `/*`

When a user inputs a line, the first few characters are checked for the EOF string, /*, which denotes the end of a read.

For example:

```
/*  ! This is a generated EOF
 /* ! This is a single argument input string "/*".
```

D.4.2 Matching keyword arguments

A common type of argument that a program might expect would be a character string that is matched against a list of possible keywords, such as a list of commands to tell the program what to do next. In general, case is not significant for such matches, (i.e., ‘KEYWORD’ would match ‘keyword’). Also, partial matches are usually allowed, (i.e., ‘key’ is a partial match to ‘keyword’. If the list of allowed matches includes two such partial matches, then usually the first in order is taken. For example if the match-list is (‘keyword’,‘keyhole’,...) then ‘key’ matches ‘keyword’. Of course, any exact match takes precedence, in (‘keyword’,...,‘key’) then ‘key’ matches ‘key’.

The particular program in use may modify any of these conditions individually. Case may be made significant, partial matches may be disallowed, or, if allowed, then only unique partial matches (so that ‘keyh’ would be necessary for a legal argument matched against the list (‘keyhole’,‘keyword’,...)).

Some programs will use a special-choice subroutine, which will prompt a user to input a single argument that will be compared with a list of choices. An example of such an interface might be as follows:

```
Please choose a keyword: (default)
```

where the string in parentheses is returned as the default if the user just returns an empty line. If the user returns a ?, then a list of all the legal options will be displayed. An EOF should be handled by the program as an exceptional reply (e.g., to revert to a previous stage of the program).

An even more special case is when the user is prompted to provide an answer to a question:

```
Do you know how to respond to this? (y)
```

In these cases, the possible answers are ‘yes’ and ‘no’. Again, the default value is in the parentheses.

Appendix E

Revision History for Version 10 & 11

E.1 Version 11.1 Changes

The following list summarizes, for reference, the major changes to XSPEC from version 11 to 11.1.

- Improvements in Data Input

  → Spectral files can now have vector values of AREASCAL and BACKSCAL. XMM-RGS spectra have an AREASCAL for each channel. The vector BACKSCAL allows the grating spectrum to be extracted from a non-rectangular region on the detector. Since AREASCAL and BACKSCAL can be vectors, they are no longer prompted for in `fakeit.`

  → The `response` command has been fixed so that XSPEC only reads the response(s) required instead of the responses corresponding to all data sets loaded. The response reading also makes more efficient use of memory, although it is still slow for large response files.

  → The constraint that the background file must have the same grouping as the source has been relaxed. Unless the file is in the deprecated SF (non-FITS) format, the background file grouping does not matter since the source file grouping is automatically used.

  → FITS NULL values are now trapped and replaced by $1.E-32$, which should have no practical effect because channels with NULL values will presumably have bad quality set.

- Model Fitting

  → The command `model ?` no longer removes the current model.

  → The Cstat statistic has been extended to work in the case where a background spectrum has been read in. The mathematical basis for this is described in Arnaud, 2001, ApJ submitted.

  → A new model type [acnmod] has been added for convolution on the model times the effective area. This is used for pile-up models in CCDs.

- Improvements in Output

  → New options on the `identify` command:

    - The user can now switch between line lists.
    - The APEC line list has been added.
    - The APEC option has additional options to specify a temperature and an emissivity limit, and uses the same APEC line list file as the apec model.

- If the command `setplot wave` has been issued, then the input range is specified in Angstroms, and wavelengths rather than energies are written out.

→ The model rate has been added to the `tclout rate` option.

→ The number of significant figures written out by the `show pha` command has been increased.

→ The `cosmo` command now allows a cosmological constant to be set. This is only allowed if the Universe is flat (when an analytic expression is available for the luminosity distance).

- Plotting Improvements

→ The `setplot id` command adds line IDs to the plots; the APEC line list is used and the temperature and emissivity limit can be set.

→ The `plot icount` command makes an integrated counts plot, useful for spectra with very few counts.

→ A new PLT feature has been used to slave any x-axis changes in the lower window to those in the upper for two-window plots. We have also explicitly included a rescale x command to ensure that the x-axis range corresponds to the data present. This overrides a PLT/PGPLOT feature that tends to produce lots of blank space on log axis plots which cover a small range.

- New Models

+ `xion` performs reflection from an ionized disk.

+ `projct` a mixing model that performs 3-D to 2-D projection for prolate ellipsoids.

+ `reflect` a convolution model for reflection from a neutral medium. It subsumes the pexrav and bexrav models.

+ `lsmooth` convolution model smoothes using a Lorentzian kernel.

+ `tbabs,ztbabs,tbgrain,tbvarabs` The Tuebingen-Boulder Galactic absorption models provide better models for absorption due to the ISM. tbabs calculates absorption including molecular gas and dust. ztbabs is a redshifted version without the dust. tbgrain allows the atomic to molecular gas ratio and the grain parameters to be varied. tbvarabs allows all parameters including gas and dust metal abundances to be varied.

- Enhancements of Current Models

→ `apec,vapec` The apec and vapec models have been sped up by a factor of 50-100. The model now uses the APEC v1.10 data files. The XSPEC_APEC environment variable has been added to enable the user to switch the input files in use.

→ `comptt` The default ranges for comptt have been altered to better reflect the physical range of validity of the model.

→ `gsmooth` A second parameter has been added to gsmooth to allow the width to vary with energy in different ways.

→ `redden,uvred` The redden and uvred models have been changed to have unit transmission shortward of the Lyman limit. This is physically wrong but allows them to be used in concert with photoelectric absorption models.

→ `nei family` The NEI models have slightly updated atomic physics. There is somewhat updated data for the Fe Kalpha line and a couple of bugs have been fixed. There are better collision strengths for He-like S, Ca, and Fe (and minor modifications for He-like ions for several other elements).

E.2 Version 11 Changes

The following list summarizes, for reference, the major changes to XSPEC from version 10 to 11.

- Various changes and bug fixes to the user interface have been made

  → interrupt (Ctrl-C) handling has been added for interactive sessions. Ctrl-C no longer terminates the program except during script execution.

  → In version 10, XSPEC could execute any valid XSPEC/tcl command while processing user input from interactive commands such as `newpar`, or `fakeit`. This could leave the program in a peculiar undefined state (for example, if the user typed 'fit' instead of giving parameter values). This possibility has now been excluded.

  → XSPEC will now correctly process multiple line tcl commands (e.g. for {set $i 1} {i < 5} {incr i} { ...) from the command prompt.

  → Changes in script processing: XSPEC scripts should be invoked with `@` rather than `source`, and should be invoked from the unix prompt with `%xspec - <script>` rather than `%xspec <script>`. The functionality is unchanged apart from the need to add an `exit` command to scripts if the program is to terminate after executing the script.

  → A new command, `tclout`, is available to return xspec internal information to the tcl interface. The command does not require the `$xs_return_result` variable to be set. tclout writes its output to a Tcl variable called `$xspec_tclout`, which can then be manipulated using Tcl commands. The v10.0 method of returning variables to the tcl interface is still supported but is deprecated.

  → The single character abbreviations for XSPEC commands have been withdrawn owing to many clashes with commands recognized by tcl's command interpreter. XSPEC commands must be at least two characters in length.

  → The command `renorm` can no longer be abbreviated to `ren`, for similar reasons.

  → During parameter setting, XSPEC now displays a text string prompt indicating the data group number (if there is more than one data group), model component, and parameter name.

  → XSPEC now loads the Tk graphical toolkit library at startup. Scripts for XSPEC can therefore be written using tcl and tk commands. For more information about tcl/tk, see

  `http://www.scriptics.com`.

  → XSPEC now sets up a directory, $XSPEC_HOME, in which the user can place tcl/tk scripts that are loaded on start up. This allows users to create their own composite command procedures in tcl/tk as augmented by XSPEC commands. Some examples are provided in the distribution: see also the online documentation at

  `http://xspec.gsfc.nasa.gov/xanadu/xspec/tcl/tcl.html`.

  $XSPEC_HOME defaults to $HOME/.xspec

- Changed default Photoelectric absorption cross-sections. The set of photoabsorption cross-sections used by default has been updated. Thus, XSPEC 11 will *not* in general reproduce results from earlier versions of the program. Those results *can* be duplicated, however, by resetting the cross section table using the new `xsect` command. See below for details.

- Y2K compliance. XSPEC had only one substantive Y2K related issue, which arose when creating data simulations (the `fakeit` command). This version produces fake data files in FITS format with a `DATE` keyword compliant with the FITS standard for files created after January 1, 2000.

- The build instructions have been completely rewritten. To build the XANADU package, extract the package from the tarfile, change to the BUILD_DIR directory, and issue the commands

  ./configure

  ./hmake

  ./hmake install

- The implementation of user models is now performed by the construction of a shared library. This will enable users to compile their own user model libraries and use them with a system-wide XSPEC installation rather than having to build and maintain a personal copy of the program. See Appendix C for detailed instructions.

- New commands and techniques:

  → New `extend` command extends the energy range over which a model is calculated. This is useful for convolution models.

  → Bayesian inference. The `bayes` command sets up Bayesian inference including specifying priors for model parameters. This is valid for data with background even when both source and background spectrum have too few counts to be in the gaussian regime.

  → Included Akaike and Bayesian Information Criteria when running C statistic.

  → Line identification. The `identify` command lists possible line identifications in the energy range requested.

  → Genetic algorithm (`genetic` command). A genetic global optimization scheme has been added to the methods. This is slow - with three free parameters it requires 1000s of generations to find the minimum. The current population can be plotted using the `plot genpop` option.

  → New `ftest` command calculates F-statistic and probability.

  → Absorption cross-sections. `xsect` command changes the photoelectric absorption cross-sections used in all the absorption models with the exception of wabs. The options are the old Balucinska-Church & McCammon cross-sections, the same but with new He cross-section, and a compilation by Verner et al.

  → Algebraic models. The `mdefine` command lets the user define a new model component as an algebraic expression. This new component is evaluated by an interpreter so will be slightly slower than installed models. However, any `mdefine`'d model component is likely to be relatively simple so the slower evaluation speed should not be significant.

  → XSPEC now auto-loads Tcl scripts to run the commands `addline` and `modid`. The former adds lines to the current model in an optimum fashion. The latter attempts to find IDs for all line components in the current model.

- → Uncertainties associated with the model. XSPEC v11 has the capability to include an uncertainty in the model. This is added in quadrature with the uncertainty in the data.

- Modified Commands:

  - → `plot` command has new option, `plot genpop`, as above.
  - → The default `steppar` option is ”best” rather than ”current”.
  - → The `error` command allows the user to restart the calculation if a new minimum is found. This can be made an automatic option.
  - → `setplot rebin` can take a plot group argument to allow plots to be rebinned independently.
  - → For OGIP Type II files, the `data` command now accepts ranges and wildcards allowing the user to read multiple spectra with a single command, e.g. `data file1{1-10}` will read the first ten spectra in file1, while `data file1{*}` will read all of the spectra in file1. The limit to the number of datasets that can be read into the program has been increase from 100 to 500.
  - → Parameter linking now allows operations of the form `newpar 4 = 3 * a + b` where a and b are real numbers. Both / and – are also allowed, e.g. `newpar 4 = 3 * 2/3 - 1.5`
  - → `show free` option shows only free parameters.
  - → Convolution components now operate on model*area rather than just model. This allows pile-up to be modelled as a convolution component.
  - → Modified `goodness` so it works for spectra with background and for the chi-squared statistic provided the only source of variance is counting statistics.

- The following new models have been added:

  + `apec, vapec`
  + `bexrav, bexriv`
  + `bmc`
  + `equil, vequil`
  + `lorentz`
  + `nei, vnei, gnei, vgnei`
  + `npshock, pshock, vnpshock, vpshock`
  + `redden`
  + `sedov, vsedov`
  + `srcut,sresc`

- The following models have been removed:

  – `tsabs`
  – `tita_a`

- The description of the normalization for all collisional plasma models in terms of luminosity distance was incorrect. This has been changed to angular size distance and there should now be the right number of (1+z) factors.

- The `model.dat` and `lmodel.dat` files have been simplified and `msetup` made more robust. For the change to `lmodel.dat` see the example file in the release. `lmodel.dat` and the local model source files can be placed in any directory. The XSPEC make procedure uses the environment variable LMODDIR to find the directory containing local models.

E.3 Version 10 Changes

The following list summarizes, for reference, the major changes to XSPEC from version 9 to 10.

- Tcl now provides the user interface. This should be mostly transparent to the user except that more options are available (see appendix D for details). There are a few changes that are required
  - `exec command` or `command` should be used instead of `$ command`.
  - `xhistory` should be used rather than `history`.
  - `recall` and `reexecute` have been replaced by `history` and `!n` (to reexecute command n).
  - The XCOMS environment variable no longer works as a list of directories in which to search for script files.
  - The XSPEC initialization script is called .xspecrc and must reside in the user's home directory.
  - Since the PLT commands are not yet integrated into the Tcl environment it is not possible to write a script containing `iplot` and then some PLT commands. Instead, use `setplot com` and `plot`.
  - The `setplot` command cannot be abbreviated to `set` but must be at least `setp`.

- Dynamic memory is now used. This means that
  - XSPEC now requires much less memory on start-up and the amount of memory used will depend on the sizes of the datasets and response matrices read in.
  - Most of the arbitrary limits on the sizes of files have been removed.
  - Some sections of XSPEC have been sped-up.

- There is a new, improved makefile which does incremental builds correctly. To force a complete new build use `xmake full`.

- The response matrix reading has been updated to use the TLMIN4 keyword in the RESPONSE extension (mainly for use with XTE HEXTE files).

- If `setplot wave` has been specified and an `ignore` or `notice` command is given with real numbers then these specify wavelengths in Angstrom. For `setplot chan` and `setplot energy` the real numbers represent energies in keV.

- The algorithm for the `error` or `uncertainty` command has been improved and should be faster and more reliable.

- A bug that caused incorrect estimated parameter uncertainties for the C-stat has been fixed.

- At the end of a fit run XSPEC now writes out the principal axes of the error ellipsoid. This can give some clues about which parameters are correlated.
- All parameters have units specified.
- XSPEC now automatically writes out its current state to the file xautosav.xcm. The default is to write out the state after every command. The `autosave` command can change the frequency of updating of xautosav.xcm or turn it off completely.
- The `editmod` command enables the user to add, delete, or replacement any one component in the model.
- The `dummyrsp` command has an additional argument which allows the creation of a virtual response matrix with perfect resolution and efficiency.
- The `abund` command has two new options. `abund aneb` uses the Anders & Ebihara (1982) relative abundances and `abund file filename` reads a set of relative abundances from `filename`.
- The extra page is no longer produced when specifying PostScript devices using the XSPEC `cpd` command.
- Three new `plot` options are available :
  - `eemodel` plots the model in νf_ν space.
  - `eeufspec` plots the unfolded spectrum in νf_ν space.
  - `dem` plots the differential emission measure distribution for the last multiple temperature model calculated.
- The slow plotting of spectra with many channels has been fixed.
- `plot summary` now works correctly and `plot eff` has wavelength as the x-axis if `setplot wave` is used.
- The algebraic format for specifying models is the default. It is now possible to specify additive components outside the overall multiplicative group. The algebraic format is used internally by XSPEC so in some cases the actual order of components and hence parameters will be different. A tool is available to convert v9 scripts into v10 scripts.
- The `addcomp` command has been modified to reflect the change in the model format.
- Table models must now be specified as e.g. `atable{filename}` instead of `atable filename`.
- A new type of model is available. Convolution models are generalizations of multiplicative models and allow more complicated operations than simple multiplications. A gaussian smoothing model (`gsmooth`) is included as an example.
- The following new models have been added and old ones improved:
  - ⋆ `grbm` is a gamma-ray burst spectrum (after Band).
  - ⋆ `c6mekl`, `c6pmekl`, `c6vmekl`, and `c6pvmkl` supersede `cp6mkl` and `cp6vmkl`.
  - ⋆ `laor` is an emission line from a disk around a black hole.

- ⋆ `plcabs` is an approximation for transmission through Compton-thick matter.
- ⋆ `gsmooth` is a convolution model to modify the current model by a Gaussian smoothing.
- ⋆ `vphabs` is a photoelectric absorption with variable abundances.
- ⋆ `zvphabs` is a redshifted photoelectric absorption with variable abundances.
- ⋆ All mekal based models have a new parameter called switch which sets whether the model spectrum is interpolated from a table or calculated (former is faster, latter is more accurate). The interpolation option can make up to a factor of 100 difference in speed.
- ⋆ `nteea`, `pexrav`, `pexriv`, and `tita_a` have all been improved.
- ⋆ `pliref`, `plrefl`, and `zplrefl` have been removed since they were inaccurate. `pexrav` and `pexriv` should be used in their place.
- ⋆ `mkcflow` and `vmcflow` use the mekal rather than meka model.

- The `model.dat` and `lmodel.dat` files have been simplified and `msetup` made more robust. For the change to `lmodel.dat` see the example file in the release. `lmodel.dat` and the local model source files can be placed in any directory. The XSPEC make procedure uses the environment variable LMODDIR to find the directory containing local models.

E.4 XSPEC v11.1 issues fixed in 11.2

11.1.0a On some Linux systems garbage characters are generated when fitting certain models (eg `zpow`, `refsch`). This is due to uninitialized status variables in a number of routines.

11.1.0b The file written by WENV after IPLOT CONT cannot be read back into QDP.

11.1.0c There is a problem reading BACKSCAL and AREASCAL columns from type II format spectral files.

11.1.0d There is a very rarely triggered bug in the `error` command which causes a crash.

11.1.0e Under Linux the data syntax `filename{number}` or `filename{*}` does not work.

11.1.0f XSPEC will crash if asked to re-open an autosave file in a directory to which the user does not have write permission.

11.1.0g Under Linux the `pexrav` model can cause the fitting process to fail (with warning messages from svdcmp).

11.1.0h The `pileup` model produces subtly wrong answers if there is pile-up beyond 2 photons.

11.1.0i The `sedov` model doesn't work under some Linux systems.

11.1.0j Under Linux several of the NEI models cause segmentation faults under some circumstances.

11.1.0k The default parameter limits on the `mkcflow` and `vmcflow` models are wrong and can cause XSPEC to crash. This is caused by a combination of incorrect parameter limits and coding errors.

11.1.0l Building XSPEC without local models and then setting the LMODDIR environmental variable to link in local models does not work.

11.1.0m If the dataset has vector areascal or backscal these are not treated correctly by the `ignore` command.

11.1.0n If the dataset has vector areascal or backscal and it is grouped then the final bin will have incorrect areascal and backscal values.

11.1.0o `Fakeit` fails for a file with vector values of more than two of quality, grouping, areascal, or backscal.

11.1.0p No warning is generated if an ARF has fewer energy ranges than the RMF with which it is paired. Technically this isn't a bug but it seems to be quite easy to create Chandra ARF/RMF pairs with this problem - the diagnostic is that the gain appears to be wrong.

11.1.0q `fakeit` does not allow its output file to be saved into a directory specified by an absolute path (i.e. beginning with a "/").

11.1.0r (user interface patch) XSPECs internal strings within the data command are not properly initialized. Additionally, buffer overflows can cause unexpected crashes on some platforms. Also, XSPEC scripts that refer to non-existent files do not properly prompt the user for a replacement filename.

11.1.0s XSPEC with MINUIT does not work under Redhat Linux 7.1.

11.1.0t The `pileup` model parameters do not vary when fitting.

11.1.0u The Anders and Grevesse Cl, Cr, and Co abundances are incorrect.

11.1.0v There is a bug in the `notice` command that is triggered if channels are noticed in a dataset and there are subsequent datasets which do not have channels noticed. eg if there are 3 datasets read in and the first has channels ignored then a `notice 1:1-**` will trigger the bug and corrupt the responses of the second and third datasets.

11.1.0w There are errors in the background spectrum contribution to the calculation of χ^2 for the non-standard weighting options (`gehrels`, `churazov`, `model`).

11.1.0x Contrary to the impression given by the help, the Tuebingen absorption models do not use the ISM abundance ratios given in the Wilms et al. paper but use the standard ratios set by the `abund` command.

11.1.0y The help on the `cosmo` command is incorrect in the case that a non-zero cosmological constant is given. It is not true that q_0 must be 0.5. Infact q_0 is ignored and it is assumed that the universe is flat with $\Omega_{matter} = 1 - \Omega_\Lambda$. The workaround is to leave $q_0 = 0.5$ and forget about it.

11.1.0z The nei models seg fault under Linux if the response energies extend outside the range of validity of the model.

11.1.0aa The normalization of the `redge` model can depend on the size of the energy bins in use.

11.1.0ab The `mkcflow` model varies the He abundance in parameter 3 along with the other elements. This is inconsistent with other collisional plasma models which fix He abundance to Solar (except for those models with all elemental abundances as free parameters).

11.1.0ac XSPEC crashes under Linux if the same type II spectral file is read using the {} syntax to specify the spectrum more than once in an XSPEC session.

11.1.0ad The `pileup` model produces subtly wrong results if the energy range in the response does not start at zero.

11.1.0ae A subtle bug can arise when reading a new dataset into XSPEC to replace a current dataset. If both datasets have the same number of ungrouped channels and the same number of grouped bins then XSPEC will apply the grouping array from the original dataset to the new one. This may not be the correct behaviour. The ignore status will also be copied from the old file to the new one. This also may not be correct.

11.1.0af The `error` command can get stuck in an infinite loop.

E.5 XSPEC v11 issues fixed in 11.1

solaris-gcc-build XSPEC does not build with gcc owing to an internal compiler error

11.0a Xspec will SEGV and dump core if parameter linking expressions are entered in Reverse Polish Notation rather than Algebraic Notation

11.0b The Xanadu/HEAsoft help system may cause a SEGV if help is entered twice during a session.

11.0c,d (c) The `fakeit` command ignores auxiliary response files (d) The `fakeit` command does not process correctly within a tcl script.

11.0e Methods for using xspec interactive commands such as `model`, `fakeit`, and `newpar` inside tcl procedures are not documented.

11.0f XSPEC will SEGV if an attempt is made to invoke a script using the syntax within another script invoked with the syntax.

11.0h When a negative multiplicative factor is used to link parameters, if the model is saved, the save file will read the factor back as an offset, not a multiplicative factor.

11.0i Invalid filename entered for table model causes SEGV

11.0j XSPEC does not recognize correct OGIP / 1992a files (as indicated by the PHAVERSN keyword) if the file is lacking the HDUCLASS keyword

11.0l Spurious scaling factors are generated when reading models back from“save” files.

11.0m XSPEC will SEGV if the character is typed in on a line by itself.

11.0n XSPEC goes into infinite loop if an incorrect plotting device is entered. Noted: 2000 April 6.

11.0o The URL on the PGPLOT ”splash” screen for the bugs page is out of date.

11.0p The entry prompt for response files in `fakeit` ignores files given by absolute paths.

11.0q `tclout` error sets `$xspec_tclout` to garbage values.

11.0.1a `tclout plot` option does not process correctly.

11.0.1b Xspec crashes (SEGV) after processing a tcl loop within a command (.xcm) file

11.0.1c The meka model shows no emission lines as it uses an input file which recent versions of cfitsio cannot read. This model is in any case obsolete and should only be used for historical comparisons.

11.0.1e The `cemekl` and `cevmkl` models do not integrate over a range of temperatures as claimed.

11.0.1g XSPEC crashes (SEGV) intermittently in the dialogue that prompts for replacement filenames.

11.0.1h XSPEC crashes (SEGV) when saved model expressions of sufficient length are read back into XSPEC from an .xcm script

11.0.1j For fake datasets, count rate was output as integer if no counting statistics option selected.

11.0.1k (1) XSPEC crashes (SEGV) when attempting to load an .xcm script from within an .xcm script using either the `@` syntax or from the command line. (2) XSPEC crashes (SEGV) when saved model expressions of sufficient length are read back into XSPEC from an .xcm script

11.0.1l (Model string ”jumbo patch”) XSPEC underallocates memory for the strings describing extremely complex models.

11.0.1m The `apec` and `vapec` models contain an error that leads to subtly incorrect spectra.

11.0.1n The F value is incorrect if more than one parameter has been added.

11.0.1o The model string printed by the show command is not properly re-initialized when the model is changed.

11.0.1p XSPEC crashes (SEGV) if user requests a fake dataset without us. If the user supplies insufficient arguments to fakeit from a script, XSPEC will now exit with an error (prior to this it would SEGV).

11.0.1q XSPEC will not perform fits if all channels are ignored in any dataset loaded.

11.0.1s XSPEC will crash if a model is defined and the last dataset read has no response matrix.

11.0.1t The `tclout rate` option did not work correctly.

11.0.1u The MKCFLOW model causes XSPEC to crash if the lower temperature limit is less than about 0.02 keV.

11.0.1v (1) Commands of the form `ignore 5.0-**` ignore channels with wavelengths less than, rather than greater than the specified value if `setplot wave` has been given. (2) If multiple datasets are in use, then even-numbered datasets will interpret the range as energy instead of wavelength.

11.0.1w steppar does not work correctly if stepping is performed over more than three parameters.

11.0.1x In the APED model, the helium abundance is varied with the other abundances instead of remaining constant at the Solar estimated value.

11.0.1y (1) Files with complex patterns of ”bad” channels may cause xspec to crash when the ignore command is invoked. (2) Version string updated to print patch level of running xspec version.

11.0.1z1 EBOUNDS arrays are constructed incorrectly for datasets with grouped channels if the arrays are decreasing with channel number.

11.0.1aa Fixes for fake file generation. (a) XSPEC crashes (SEGV) if more than about 100 files are simulated (e.g. from a large OGIP-Type II file). (b) simulating background causes corrupted memory if no dataset has been read in.

11.0.1ab Fixes xspec tcl scripts implementing `addline` and `modid` commands.

11.0.1ac In XSPEC v11 convolution components work on the model*(effective area). This was changed from v10 to enable pile-up to be implemented as a convolution model. However, this change can introduce features when using other types of convolution models. This patch reverts xspec to the v.10 behaviour.

11.0.1ad XSPEC crashes intermittently in the command `tclout model.`

11.0.1ae (1) XSPEC crashes on `fakeit` command if no datasets are loaded. (2) Verbose output printed to log file even when chatter level is set to 0 (3) XSPEC crashes in the replacement file dialogue for new table model if user enters a blank line. (4) some linux version builds fail to compile the xanlib library because of the lack of some trigonometric fortran functions.

11.0.1af XSPEC crashes on DEC/Compaq platform if spectra with NULL or INDEF values are present in the COUNT or RATE column, as occurs with XMM/RGS files.

11.0.1ag The `grad` model contains several bugs, causing the mass obtained from fitting the model to the observation to be over-estimated by a factor 1.4. These bugs have been fixed and a new parameter (par6) added to make clear the distinction between the old and new models. The online help has been amended accordingly.

11.0.1ah When operating in wavelengths (after setplot wave) XSPEC does not ignore correctly if and attempt is made to specify the ignore range in wavelength (eg ignore 5.0-6.0) and the input dataset is stored in increasing wavelength (e.g. XMM-Newton RGS spectra).

11.0.1ai `gain fit` option fails in command scripts.

11.0.1aj `fakeit none` causes XSPEC to halt if there are no datasets previously defined and the user enters an invalid file.

Fixed but not previously reported

Neither the ** option nor giving a wavelength outside the dataset range worked correctly when ignoring or noticing after setplot wave.

There was a spurious zero width energy bin produced by the `extend high` command.

The background scaling was not being taken into account when using the `lstat` statistic.

When using the annealing method the wrong parameters were updated if any had been frozen.

In `steppar` the best/current choice had to precede the log/nolog choice. Also, the write statement for the output for each step did not work correctly for more than 3 parameters.

The `save` command now did not save gain shift information when gain was not being fitted.

There were illegal blanks in the TFORM# keywords in the eigen*.fits for the NEI models.

In `linefile.fits` the TRANS column was A8 which failed to newer versions of cfitsio.

There was a minor typo in `photo.f.` It would only have made a difference if there was more than one edge in a single energy bin, which probably never happens.

There was an error in the table interplation algorithm which showed up if parameters had values less than about 1e-5.

An error was corrected in `comptt` that was introduced in July 99. This has been fixed along with the original problem that the earlier incorrect fix was supposed to handle.

Only the first mixing model was available.

There was a bug that gave an incorrect equivalent width for all datasets except the first if multiple datasets were in use and the lower end of the continuum range used corresponded to the lowest energy in the response.

The model `cevmkl` was actually using the C abundance as the density. This would only have mattered if a high density plasma was being analyzed.